L Number	Hits	Search Text	DB	Time stamp
1	168	(token near1 ring) and (TDM or time adj slot$1) and master and slave	USPAT; US-PGPUB; DERWENT	2004/04/01 09:01
2	98	((token near1 ring) and (TDM or time adj slot$1) and master and slave) and (control with (distribut$3 or tree))	USPAT; US-PGPUB; DERWENT	2004/04/01 09:02
-	40	TDM$1.ab. and ((receiv$3 and transmit$4) near4 (control adj data))	USPAT; US-PGPUB; DERWENT	2004/03/31 09:18
-	156	TDM$1 and control.ab. and ((receiv$3 and transmit$4) near4 (control adj data))	USPAT; US-PGPUB; DERWENT	2004/03/30 11:35
-	25	(TDM$1 and control).ab. and ((receiv$3 and transmit$4) near4 (control adj data))	USPAT; US-PGPUB; DERWENT	2004/03/30 11:24
-	4	(TDM$1 and control).ab. and ((receiv$3 and transmit$4) near4 (control adj data)) and (control with master)	USPAT; US-PGPUB; DERWENT	2004/03/30 11:24
-	325	(TDM$1 and control).ab. and ((receiv$3 and transmit$4) near4 control)	USPAT; US-PGPUB; DERWENT	2004/03/30 11:41
-	300	((TDM$1 and control).ab. and ((receiv$3 and transmit$4) near4 control)) not ((TDM$1 and control).ab. and ((receiv$3 and transmit$4) near4 (control adj data)))	USPAT; US-PGPUB; DERWENT	2004/03/30 11:36
-	24	(((TDM$1 and control).ab. and ((receiv$3 and transmit$4) near4 control)) not ((TDM$1 and control).ab. and ((receiv$3 and transmit$4) near4 (control adj data)))) and master and slave	USPAT; US-PGPUB; DERWENT	2004/03/30 11:37
-	24	((((TDM$1 and control).ab. and ((receiv$3 and transmit$4) near4 control)) not ((TDM$1 and control).ab. and ((receiv$3 and transmit$4) near4 (control adj data)))) and master and slave) not ((TDM$1 and control).ab. and ((receiv$3 and transmit$4) near4 (control adj data)) and (control with master))	USPAT; US-PGPUB; DERWENT	2004/03/30 11:36
-	9	(((TDM$1 and control).ab. and ((receiv$3 and transmit$4) near4 control)) not ((TDM$1 and control).ab. and ((receiv$3 and transmit$4) near4 (control adj data)))) and master and slave and (control near4 master)	USPAT; US-PGPUB; DERWENT	2004/03/30 11:37
-	27	(TDM$1 and control).ab. and (master and slave) and ((receiv$3 and transmit$4) near4 control)	USPAT; US-PGPUB; DERWENT	2004/03/31 09:09
-	1	(TDM$1 and control).ab. and (distribut$3 near5 control) with (multiplex$3 or mux$3) and ((pass or relay) with control) and master and slave	USPAT; US-PGPUB; DERWENT	2004/03/31 09:11
-	3	(TDM$1 and control).ab. and (distribut$3 near5 control) with (multiplex$3 or mux$3) and ((pass or relay) with control)	USPAT; US-PGPUB; DERWENT	2004/03/31 09:12
-	11	(TDM$1 and control).ab. and (distribut$3 near5 control) with (multiplex$3 or mux$3)	USPAT; US-PGPUB; DERWENT	2004/03/31 09:12
-	9	(hierarch$3 with (control near5 distribut$3)) and TDM$1 and (multiplex$3 or mux$2)	USPAT; US-PGPUB; DERWENT	2004/03/31 09:21
-	3	SDLC and (multi-point or multi-drop) and TDM$1 and (control with distribut$3)	USPAT; US-PGPUB; DERWENT	2004/03/31 09:40
-	41	(control near4 tree) and TDM	USPAT; US-PGPUB; DERWENT	2004/03/31 13:29
-	51	(control near4 tree) and TDM$1	USPAT; US-PGPUB; DERWENT	2004/03/31 09:32

-	4	((multi-point or multi-drop) and TDM$1 and (control with (distribut$3 or tree))) and (master near2 control)	USPAT; US-PGPUB; DERWENT	2004/03/31 09:41
-	141	(multi-point or multi-drop) and TDM$1 and (control with (distribut$3 or tree))	USPAT; US-PGPUB; DERWENT	2004/03/31 09:42
-	67	(multi-point or multi-drop) and TDM$1 and (control$3 with (distribut$3 or tree)) and control.ab.	USPAT; US-PGPUB; DERWENT	2004/03/31 09:45
-	6184	TDM$1 with control$3	USPAT; US-PGPUB; DERWENT	2004/03/31 09:46
-	1540	TDM$1 near2 control$3	USPAT; US-PGPUB; DERWENT	2004/03/31 09:46
-	81	(multi-point or multi-drop) and TDM$1 and (control$3 with (distribut$3 or tree)) and control$3.ab.	USPAT; US-PGPUB; DERWENT	2004/03/31 10:01
-	3304	(370/321,324,336,345,442,458,498,522,527,529,539,541).CCLS.	USPAT; US-PGPUB; DERWENT	2004/03/31 10:02
-	6	((370/321,324,336,345,442,458,498,522,527,529,539,541).CCLS.) and ((multi-point or multi-drop) and TDM$1 and (control$3 with (distribut$3 or	USPAT; US-PGPUB; DERWENT	2004/03/31 10:03
-	33	TDM and (control$3 with (insert and extract)) and (distribut$3 or tree)	USPAT; US-PGPUB; DERWENT	2004/03/31 10:06
-	6	(TDM and (control$3 with (insert and extract)) and (distribut$3 or tree)) and ((370/321,324,336,345,442,458,498,522,527,529,539,541).CCLS.)	USPAT; US-PGPUB; DERWENT	2004/03/31 10:04
-	5	((TDM and (control$3 with (insert and extract)) and (distribut$3 or tree)) and ((370/321,324,336,345,442,458,498,522,527,529,539,541).CCLS.)) not (((370/321,324,336,345,442,458,498,522,527,529,539,541).CCLS.) and ((multi-point or multi-drop) and TDM$1 and (control$3 with (distribut$3 or	USPAT; US-PGPUB; DERWENT	2004/03/31 10:04
-	201	TDM and (control$3 with (insert$3 and extract$3)) and (distribut$3 or tree)	USPAT; US-PGPUB; DERWENT	2004/03/31 10:42
-	32	(TDM and (control$3 with (insert$3 and extract$3)) and (distribut$3 or tree)) and control$3)	USPAT; US-PGPUB; DERWENT	2004/03/31 10:42
-	4	(TDM and (control$3 with (insert$3 and extract$3)) and (distribut$3 or tree)) and ((370/321,324,336,345,442,458,498,522,527,529,539,541).CCLS.) and (nodes or multiplex$3 or mux$2) same (control with (distribut$3 or tree)) and TDM and master	USPAT; US-PGPUB; DERWENT	2004/03/31 10:38
-	133	(nodes or multiplex$3 or mux$2) same (control with (distribut$3 or tree)) and TDM and master	USPAT; US-PGPUB; DERWENT	2004/03/31 10:43
-	6	((nodes or multiplex$3 or mux$2) same (control with (distribut$3 or tree)) and TDM and master) and ((370/321,324,336,345,442,458,498,522,527,529,539,541).CCLS.)	USPAT; US-PGPUB; DERWENT	2004/03/31 11:53
-	242	(distribut$3 or tree)	USPAT; US-PGPUB; DERWENT	2004/03/31 11:53
-	8	(master and slave and TDM and control with (distribut$3 or tree)) and ((370/321,324,336,345,442,458,498,522,527,529,539,541).CCLS.)	USPAT; US-PGPUB; DERWENT	2004/03/31 11:53
-	13	(hierarch$4 and sync$11 and master and slave and (TDM or (time adj slot$1))) and ((370/321,324,336,345,442,458,498,522,527,529,539,541).CCLS.)	USPAT	2004/03/31 12:03
-	328	hierarch$4 and sync$11 and master and slave and (TDM or (time adj slot$1))	USPAT	2004/03/31 12:06
-	5	hierarch$4 and (sync$11 or control$3).ab. and master and slave and TDM.ab. and (time adj slot$1)	USPAT	2004/03/31 12:13
-	180	(master near3 control) and slave$1 and TDM and process$3	USPAT; US-PGPUB; DERWENT	2004/03/31 12:15
-	14	((master near3 control) and slave$1 and TDM and process$3) and ((370/321,324,336,345,442,458,498,522,527,529,539,541).CCLS.)	USPAT; US-PGPUB; DERWENT	2004/03/31 12:28

-	7445	(control or sync$11 or clock) near4 (time adj slot$1)	USPAT; US-PGPUB; DERWENT	2004/03/31 12:29
-	152	(control or sync$11 or clock) near4 (time adj slot$1) and master and slave and TDM	USPAT; US-PGPUB; DERWENT	2004/03/31 12:42
-	50	(transmit$4 and receiv$3) with ((control or sync$11 or clock) adj data) and TDM and master and slave$1	USPAT; US-PGPUB; DERWENT	2004/03/31 12:43
-	23	(transmit$4 and receiv$3) with ((control or sync$11 or clock) adj data) and TDM and master and slave$1 and (time adj slot$1)	USPAT; US-PGPUB; DERWENT	2004/03/31 12:47
-	27	((transmit$4 and receiv$3) with ((control or sync$11 or clock) adj data) and TDM and master and slave$1) not ((transmit$4 and receiv$3) with ((control or sync$11 or clock) adj data) and TDM and master and slave$1 and (time adj slot$1))	USPAT; US-PGPUB; DERWENT	2004/03/31 12:54
-	34	multiplex$3 with (control near1 data) and TDM and master and slave	USPAT; US-PGPUB; DERWENT	2004/03/31 13:17
-	481	(propagat$3 or disseminat$3) near7 ((control or sync$11 or clock) near1 data)	USPAT; US-PGPUB; DERWENT	2004/03/31 13:19
-	16	((propagat$3 or disseminat$3) near7 ((control or sync$11 or clock) near1 data)) and TDM	USPAT; US-PGPUB; DERWENT	2004/03/31 13:19
-	16	(propagat$3 or disseminat$3) near7 ((control or sync$11 or clock) near1 data) and (node$1 or multiplex$3 or station$1) and TDM	USPAT; US-PGPUB; DERWENT	2004/03/31 13:27
-	777	SDLC	USPAT	2004/03/31 13:31
-	71	SDLC and master and slave and control adj data	USPAT	2004/03/31 14:03
-	81	(token adj ring) and TDM and master and slave	USPAT; US-PGPUB; DERWENT	2004/03/31 14:08
-	46	(TDM and ring near2 network) and master and slave	USPAT; US-PGPUB; DERWENT	2004/03/31 14:10

US006370159B1

(12) **United States Patent**
Eidson

(10) **Patent No.: US 6,370,159 B1**
(45) **Date of Patent: Apr. 9, 2002**

(54) **SYSTEM APPLICATION TECHNIQUES USING TIME SYNCHRONIZATION**

(75) Inventor: **John C. Eidson**, Palo Alto, CA (US)

(73) Assignee: **Agilent Technologies, Inc.**, Palo Alto, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

(21) Appl. No.: **09/120,402**

(22) Filed: **Jul. 22, 1998**

(51) **Int. Cl.**[7] **H04J 3/06**
(52) **U.S. Cl.** **370/503**; 370/507
(58) **Field of Search** 370/503, 507

(56) **References Cited**

U.S. PATENT DOCUMENTS

4,890,222 A 12/1989 Kirk
5,384,563 A 1/1995 Massey
5,566,180 A * 10/1996 Eidson et al. 370/94.2
6,160,819 A * 12/2000 Partridge et al. 370/474
6,236,623 B1 * 5/2001 Read et al. 368/46

OTHER PUBLICATIONS

Kopetz, Hermann et al. "Clock Synchronization in Distributed Real-Time Systems" IEEE Transactions on COmputers, vol. C-36, No. 8, Aug. 1987, pp. 933–940.

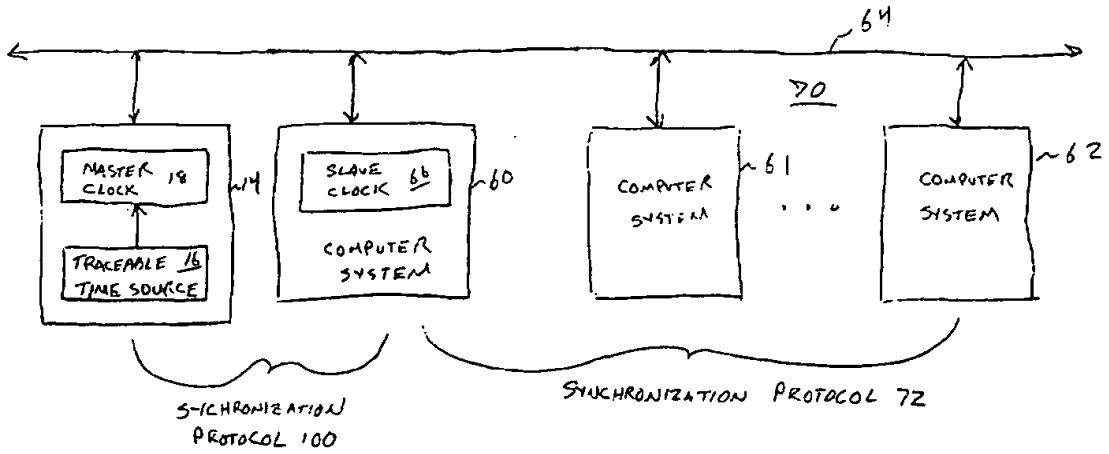
* cited by examiner

Primary Examiner—Melvin Marcelo

(57) **ABSTRACT**

A method and apparatus for accurately distributing traceable time values to a set of nodes in a system. Each node includes a slave clock that synchronizes a slave time value using a synchronization protocol. The system includes a traceable time source that generates a traceable time value and a master node having a master clock that synchronizes a master time value to the traceable time value and that distributes the master time value to the slave clocks via the communication link. The nodes may be distributed nodes or cards connected to a backplane.

18 Claims, 8 Drawing Sheets



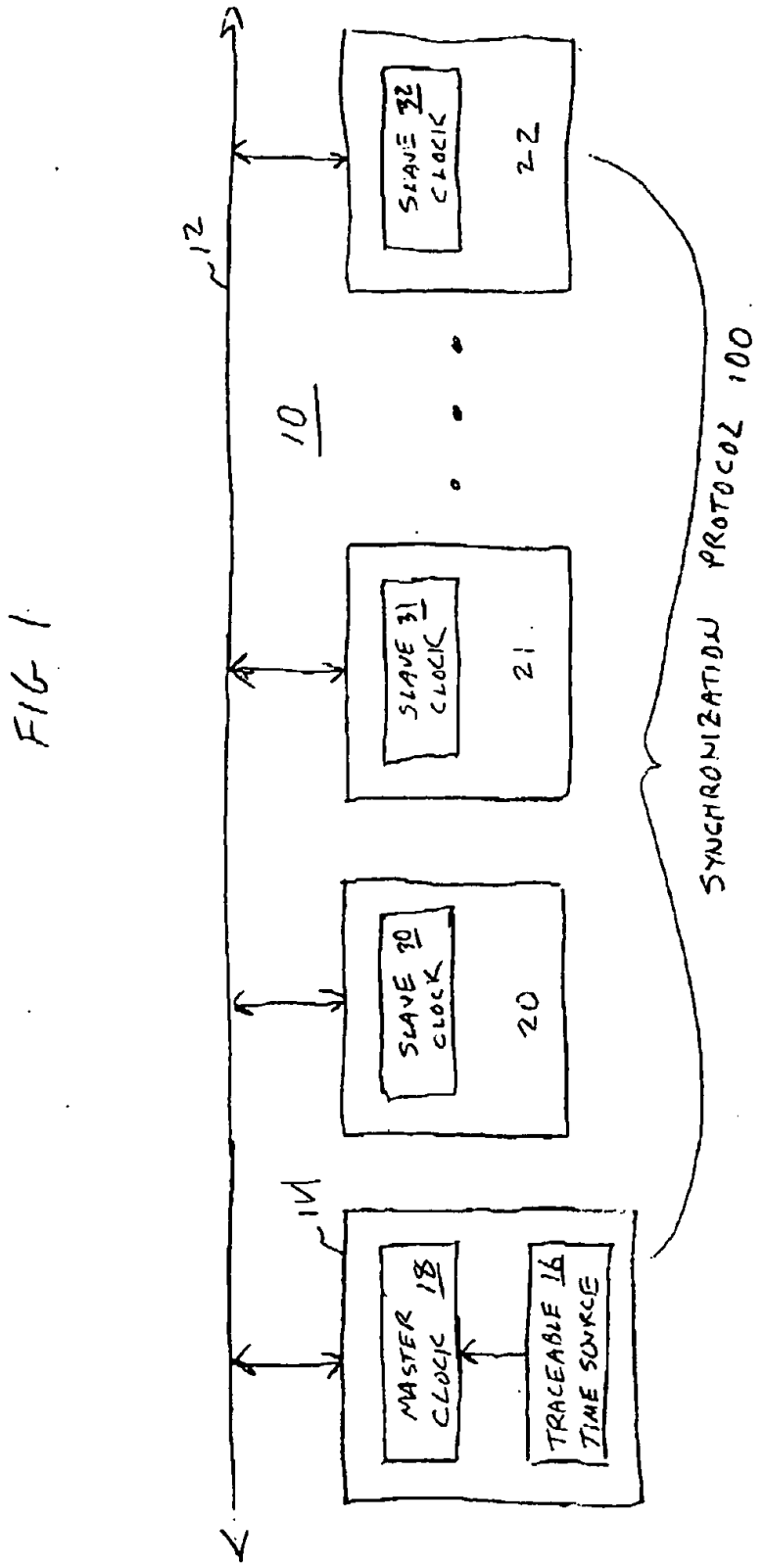
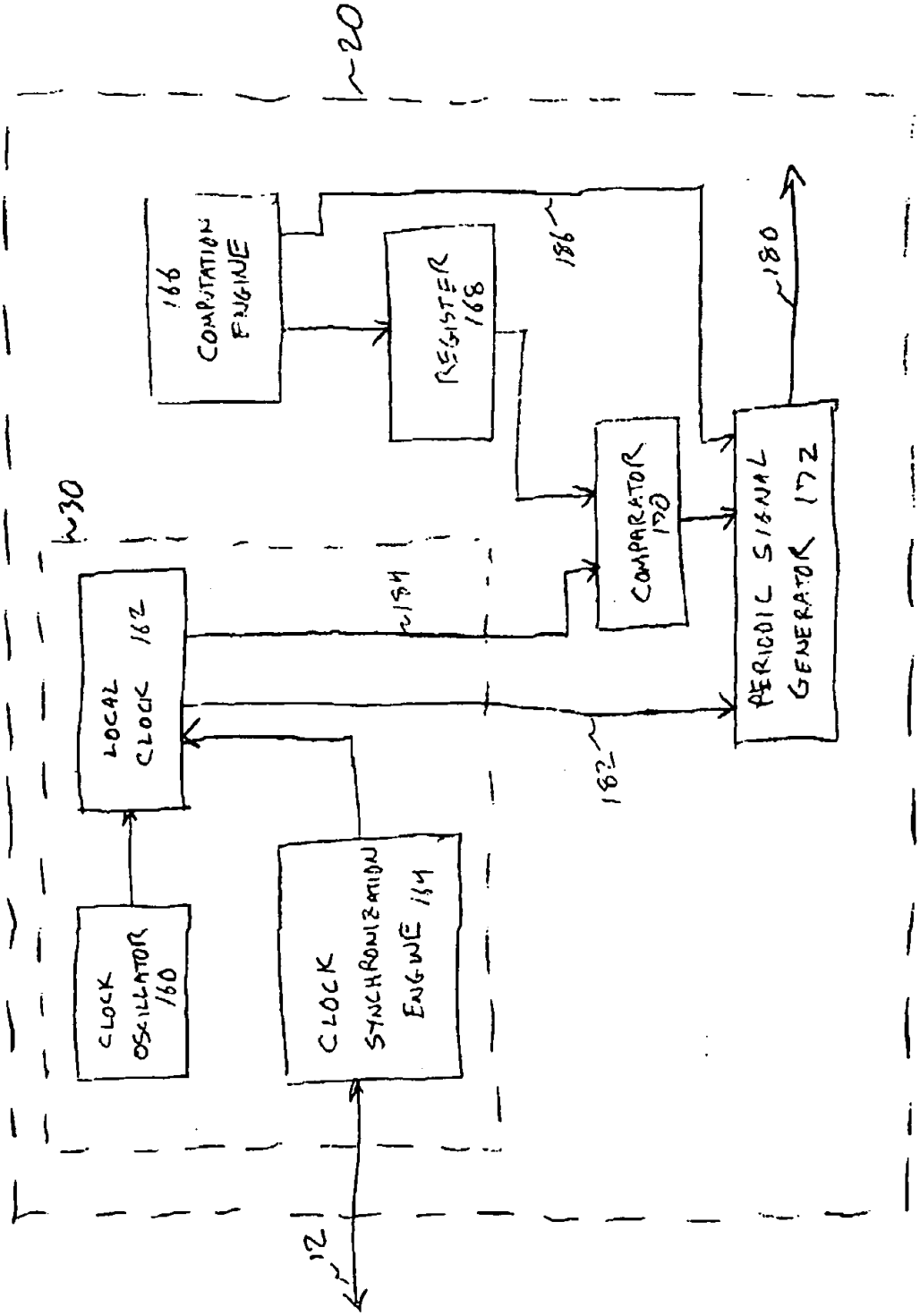

12
10
SLAVE 32 CLOCK
22
SLAVE 31 CLOCK
21
SLAVE 30 CLOCK
20
14
MASTER CLOCK 18
TRACEABLE 16 TIME SOURCE
SYNCHRONIZATION PROTOCOL 100

FIG 1

FIG 2


20
166
COMPUTATION ENGINE
REGISTER 168
186
180
30
COMPARATOR 120
PERIODIC SIGNAL GENERATOR 172
LOCAL CLOCK 162
194
182
CLOCK OSCILLATOR 160
CLOCK SYNCHRONIZATION ENGINE 174
12

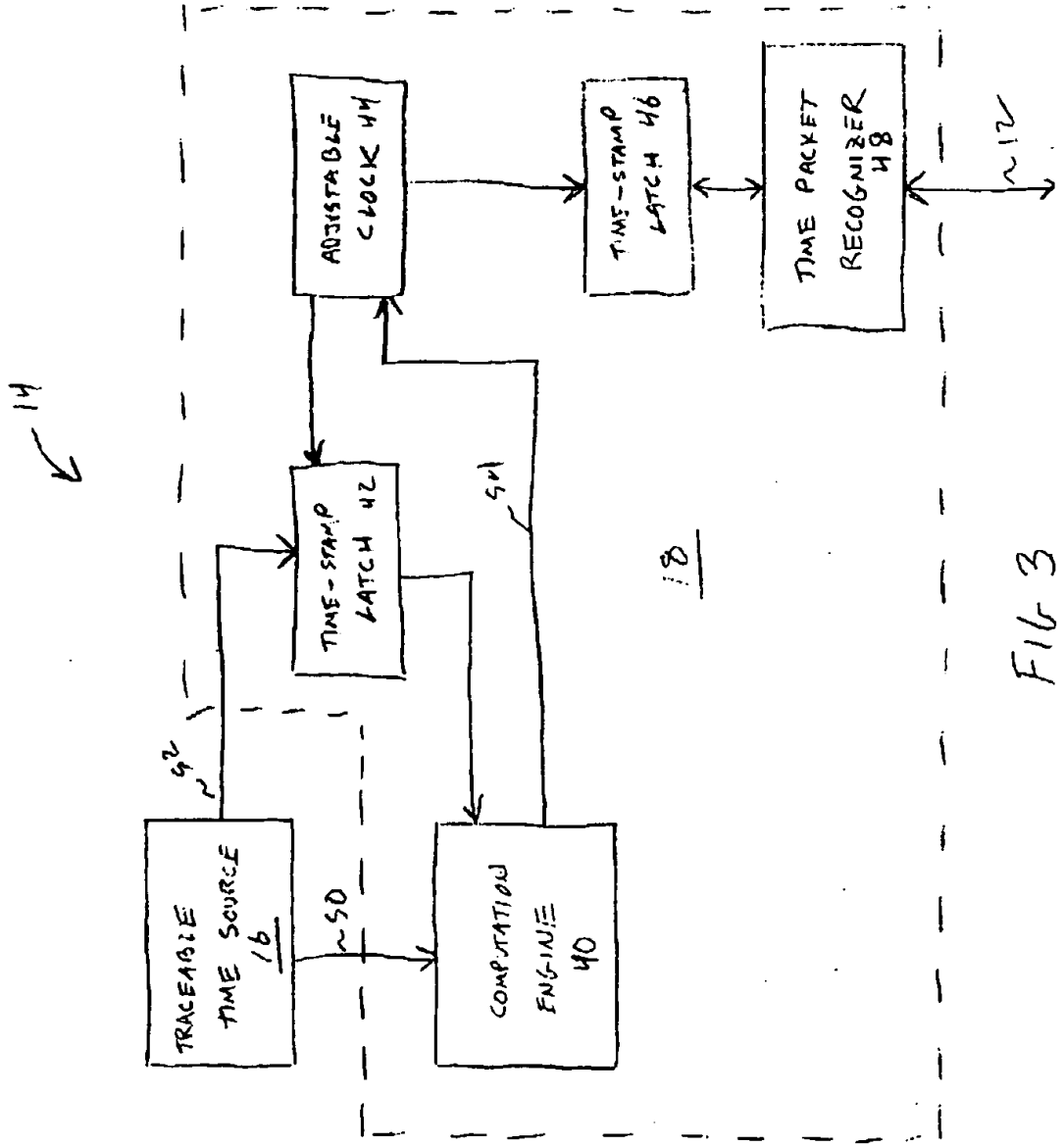

14
ADJUSTABLE CLOCK 44
TIME-STAMP LATCH 46
TIME PACKET RECOGNIZER 48
12
TIME-STAMP LATCH 42
54
18
52
TRACEABLE TIME SOURCE 16
50
COMPUTATIONAL ENGINE 40

FIG 3

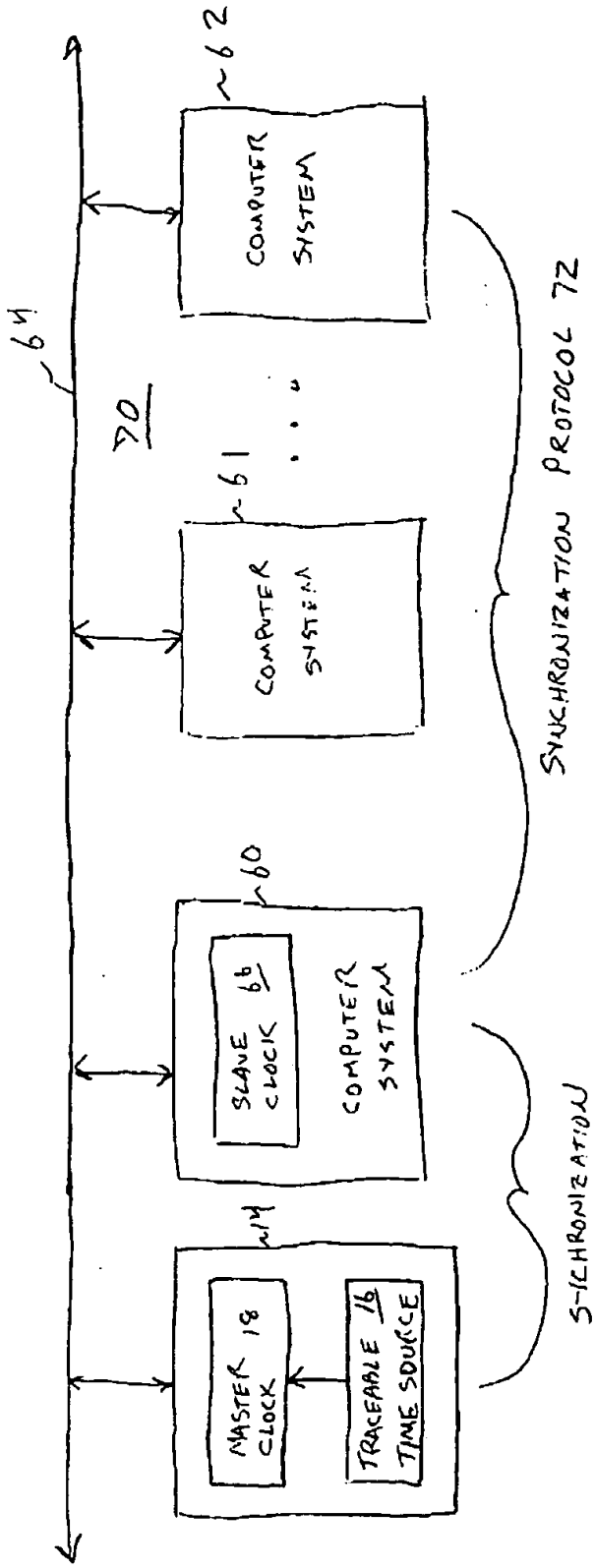

62
COMPUTER SYSTEM
64
70
61
COMPUTER SYSTEM
SYNCHRONIZATION PROTOCOL 72
60
SLAVE 66 CLOCK
COMPUTER SYSTEM
14
MASTER 18 CLOCK
TRACEABLE 16 TIME SOURCE
SYNCHRONIZATION PROTOCOL 100

FIG 4

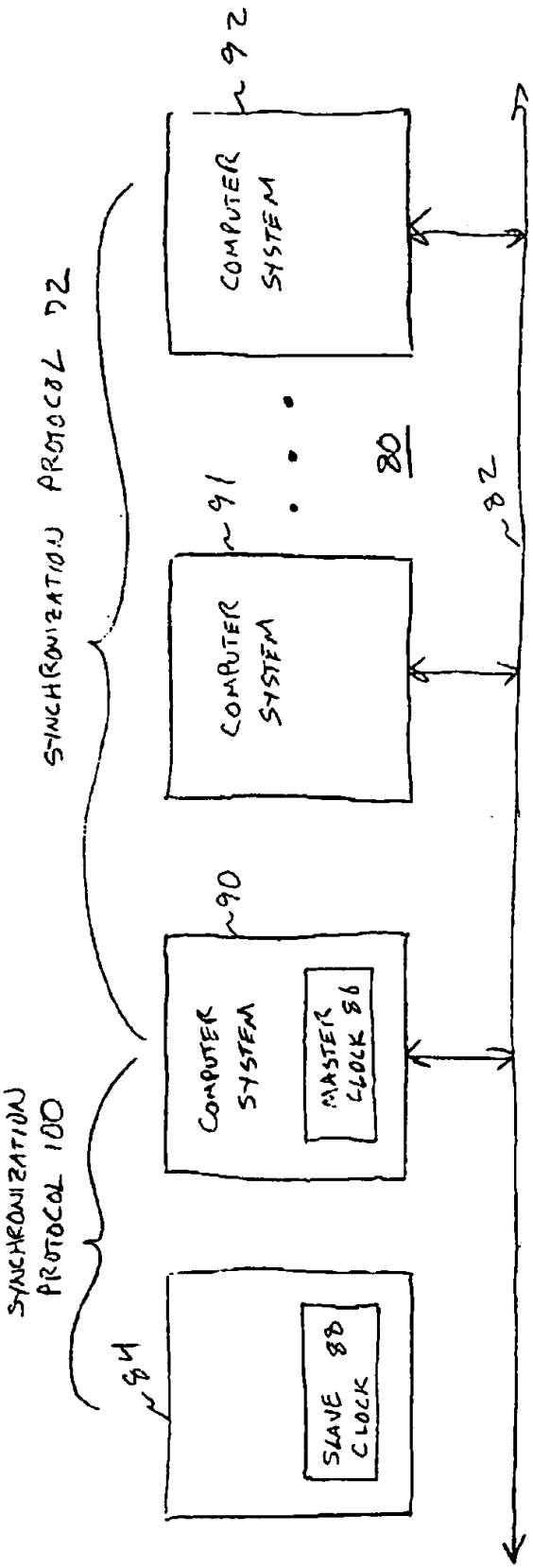
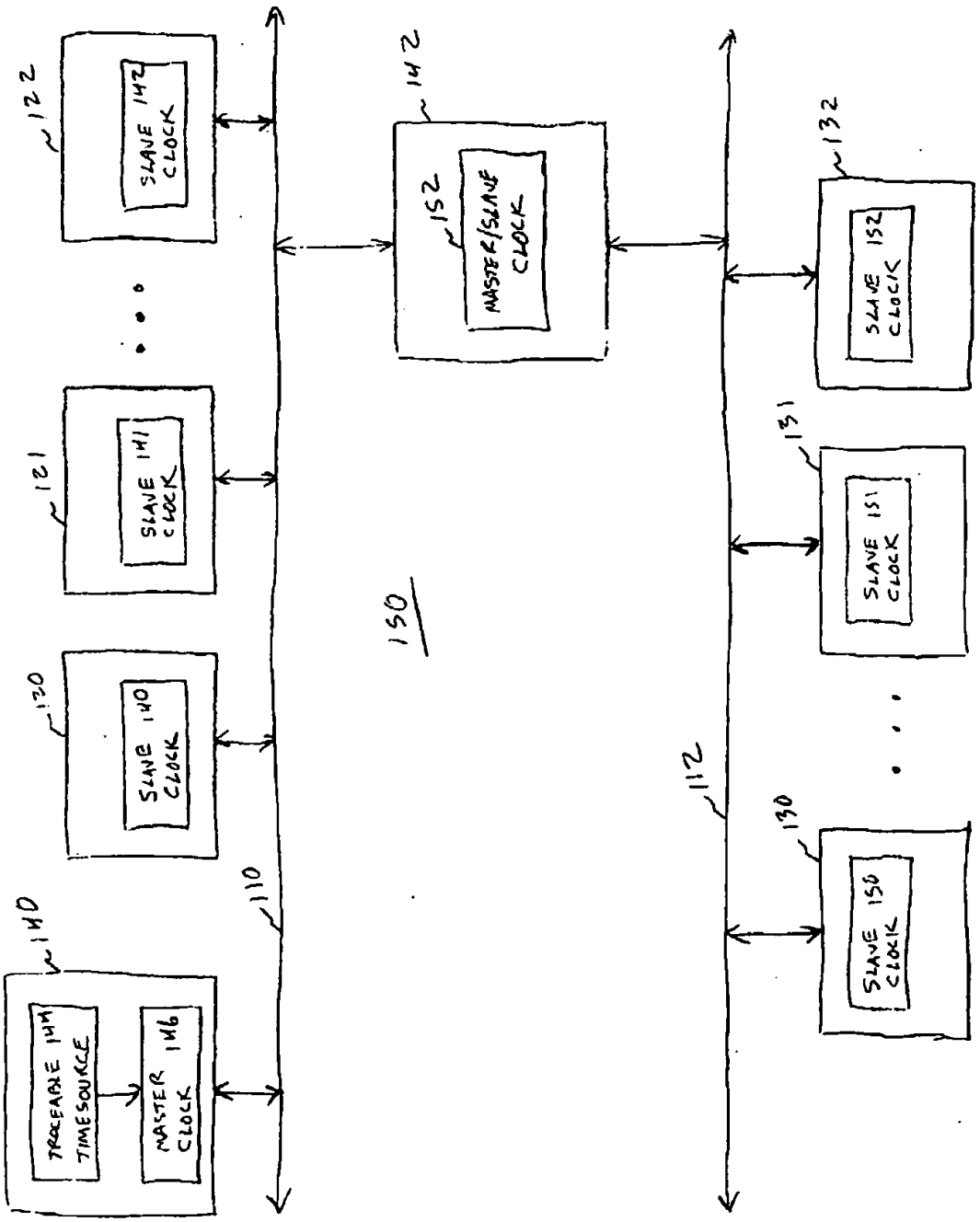

92
COMPUTER SYSTEM
SYNCHRONIZATION PROTOCOL 92
91
COMPUTER SYSTEM
80
82
90
COMPUTER SYSTEM
MASTER CLOCK 86
SYNCHRONIZATION PROTOCOL 100
84
SLAVE 88 CLOCK

FIG 5

FIG 6


150
110
112
140
TRACEABLE TIMESOURCE 146
MASTER CLOCK 146
120
SLAVE 140 CLOCK
121
SLAVE 141 CLOCK
122
SLAVE 142 CLOCK
142
MASTER/SLAVE CLOCK 152
130
SLAVE 150 CLOCK
131
SLAVE 151 CLOCK
132
SLAVE 152 CLOCK

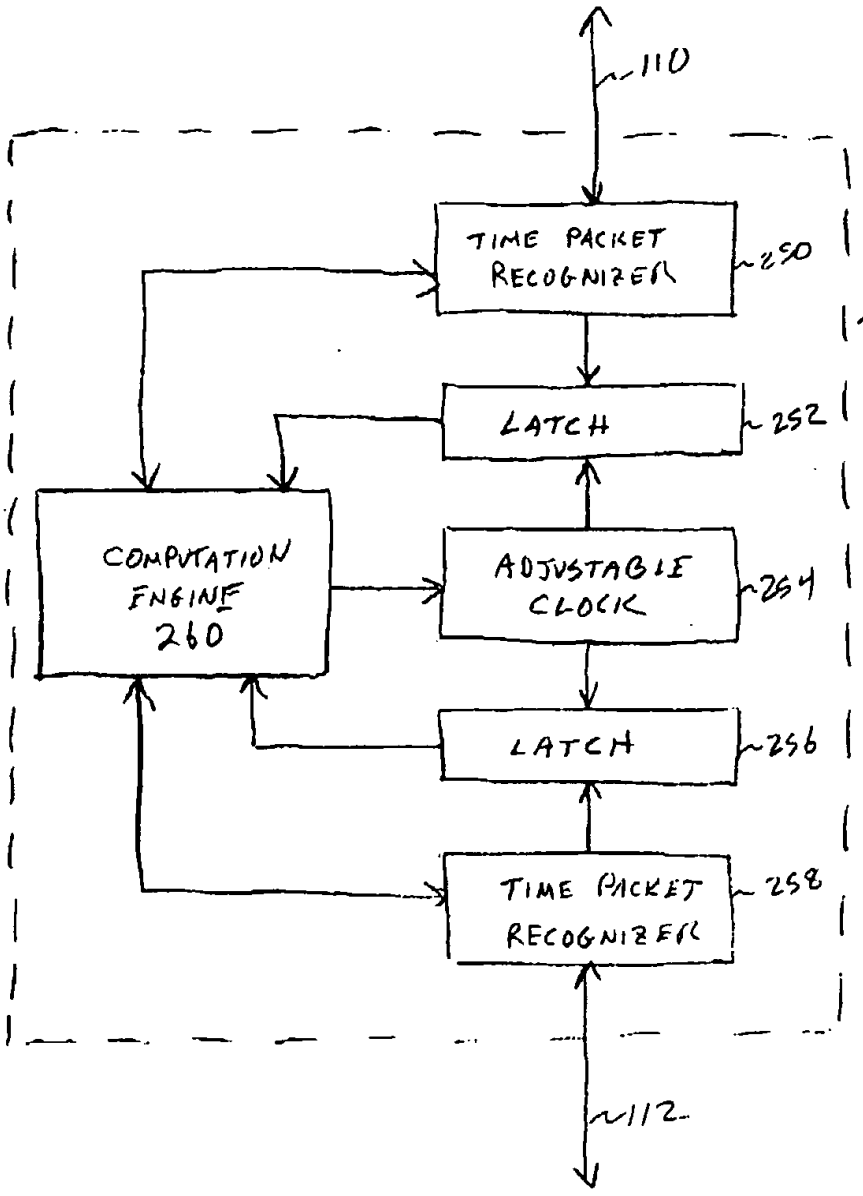

110
TIME PACKET RECOGNIZER
250
152
LATCH
252
COMPUTATION ENGINE 260
ADJUSTABLE CLOCK
254
LATCH
256
TIME PACKET RECOGNIZER
258
112

FIG 7

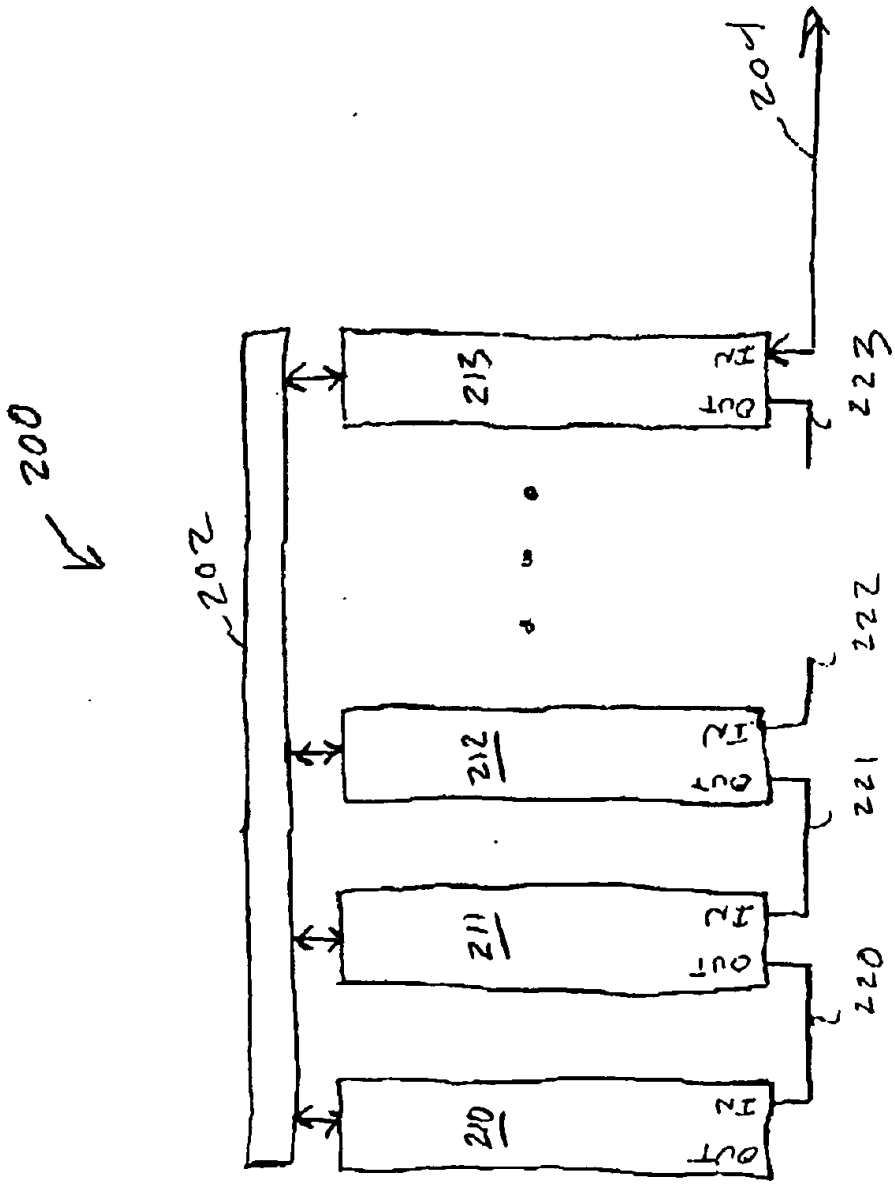


FIG 8

SYSTEM APPLICATION TECHNIQUES USING TIME SYNCHRONIZATION

BACKGROUND OF THE INVENTION

1. Field of Invention

The present invention pertains to the field of systems. More particularly, this invention relates to system application techniques using time synchronization technology.

2. Art Background

Distributed control systems are commonly arranged as a collection of nodes which are interconnected via one or more network communication links. These network communication links may be packetized links such as Ethernet or one or more of a variety of other packetized links that are adapted to distributed control system applications.

Distributed control systems commonly benefit from precise control of the timing at the distributed nodes. U.S. Pat. No. 5,566,180 of Eidson et. al. provides a method and apparatus for providing precise control of timing in distributed nodes by synchronizing the local clocks in the distributed nodes. In many applications, it may be desirable that the time kept by the distributed nodes be traceable to a standard time. Such may be useful in applications, for example, in which events in the distributed nodes must accurately occur with respect to a particular date and/or time of day.

SUMMARY OF THE INVENTION

A method and apparatus is disclosed for accurately distributing traceable time values to a set of nodes in a system. Each node includes a slave clock that synchronizes a slave time value using a synchronization protocol. The system includes a traceable time source that generates a traceable time value and a master node having a master clock that synchronizes a master time value to the traceable time value and that distributes the master time value to the slave clocks via the communication link. The nodes may be distributed nodes or cards connected to a backplane. Also disclosed are a variety of techniques for distributing the information associated with the synchronization protocol and a variety of applications for the synchronized timing in the slave nodes.

Other features and advantages of the present invention will be apparent from the detailed description that follows.

BRIEF DESCRIPTION OF THE DRAWINGS

The present invention is described with respect to particular exemplary embodiments thereof and reference is accordingly made to the drawings in which:

FIG. 1 shows a distributed system which includes a master node that distributes traceable time values to a set of nodes;

FIG. 2 illustrates a node with a waveform generator that derives a waveform period from the counter in a slave clock;

FIG. 3 illustrates a master clock with traceable time in one embodiment;

FIG. 4 illustrates a technique for introducing traceable time into a system which includes a set of computer systems;

FIG. 5 illustrates a technique for using one synchronization protocol to distribute the time values provided by a different synchronization protocol;

FIG. 6 shows a technique for distributing traceable time values to nodes which are connected to different sub-nets of a system;

FIG. 7 illustrates one embodiment of a master/slave clock in a boundary node;

FIG. 8 illustrates clock synchronization in a data acquisition and control system which includes a set of cards connected to a backplane.

DETAILED DESCRIPTION

FIG. 1 shows a distributed system 10 which includes a master node 14 that distributes traceable time values to a set of nodes 20–22. The master node 14 and the nodes 20–22 are interconnected via a communication link 12. The master node 14 includes a master clock 18 and the nodes 20–22 include a set of slave clocks 30–32, respectively.

The master node 14 includes a traceable time source 16 that generates traceable time values. A traceable time value may be defined as a time value which is derived from a standard time such as UTC time which was formerly known as Greenwich Mean Time (GMT). The master clock 18 includes mechanisms that synchronize a master time value held in the master clock 18 to the traceable time values obtained from the traceable time source 16.

The master clock 18 and the slave clocks 30–32 implement a synchronization protocol 100. According to the synchronization protocol 100, the master clock 18 and the slave clocks 30–32 exchange packets via the communication link 12 so that the slave clocks 30–32 synchronize to the master time value held in the master clock 18. As a consequence, traceable time values are accurately distributed to the nodes 20–22 using the synchronization protocol 100 because the master time value in the master clock 18 is synchronized to the traceable time values from the traceable time source 16.

In one embodiment, the synchronization protocol 100 and related mechanisms implemented in the master clock 18 and the slave clocks 30–32 are those described in U.S. Pat. No. 5,566,180. For example, each of the slave clocks 30–32 may include circuitry for adjusting its respective locally stored time value based upon computations of the sending and receiving time of time data packets which are transferred over the communication link 12. The adjustment of a locally stored time value may be accomplished by implementing a local clock in each slave clock 30–32 as a counter driven by an oscillator with sufficient stability. The least significant few bits of the counter may be implemented as an adder so that the increment on oscillator periods may be occasionally increased or decreased to effectively speed up or slow down the local clock in accordance with the results of the computation.

In one embodiment, the traceable time source 16 is a global positioning system (GPS) receiver. In other embodiments, other types of traceable time sources may be used including radio broadcast time sources such as WWV or atomic clocks.

The master node 14 and the nodes 20–22 may be any type of node in the distributed system 10. For example, any one or more of the master node 14 and the nodes 20–22 may be a sensor node or an actuator node or an application controller node or a combination of these in a distributed control system. Any one or more of the master node 14 and the nodes 20–22 may be a computer system such as a personal computer.

The communication link 12 may be implemented with one or more of a variety of communication mechanisms. In one embodiment, the communication link 12 is an Ethernet communication network. In another embodiment, the communication link 12 is a LonTalk field-level control bus which is specialized for the process control environment. In other embodiments, the communication link 12 may be

implemented with time division multiple access (TDMA) or token ring protocols to name only a few possibilities.

The synchronization protocol **100**, with or without traceable time, may be used to provide periodic phase coherent signals at widely dispersed points which correspond to the physical locations of the nodes **20–22**. For example, each of the nodes **20–22** may implement a schedule of times marking desired signal events. Each node **20–22** compares these scheduled times to the local times provided by the corresponding slave clocks **30–32**. When a scheduled event time matches the local time with appropriate accuracy, then a scheduled event is generated. The scheduled events may be sampling events wherein the nodes **20–22** include attached sensors. The nodes **20–22** may include waveform generators if analog signals are needed for events. In such a case, a match between local time and a time of a scheduled event triggers a waveform generator.

FIG. **2** illustrates the node **20** with a waveform generator that derives a waveform period from the counter in the slave clock **30**. The slave clock **30** includes a clock oscillator **160**, a local clock **162**, and a clock synchronization engine **164**. The clock synchronization engine **164** obtains timing data packets from the communication link **12** and synchronizes a slave time value in the local clock **162** using the synchronization protocol **100**. The local clock **162** provides a slave time value **184** to a comparator **170**. The comparator **170** compares the slave time value **184** to an event time held in a register **168**. The event time is provided by a computation engine **166**. When the slave time value **184** matches the event time in the register **168** the comparator **170** triggers a periodic signal generator **172** which generates a waveform **180**.

The period of the waveform **180** is controlled by a set of signals **182** from the local clock **162**. The local clock **162** includes a counter portion and a clock adjustment portion. It is preferable that the signals **182** provide bits of the counter portion of the local clock **162** which are more significant than the lower order bits of the local clock **162** used for the clock adjustment portion. The periodic signal generator **172** may be implemented using a digital phase locked loop which is driven by the signal **182**, or using other well know techniques.

The synchronization protocol **100**, with or without traceable time, may be used to provide the basis for time division multiplex access (TDMA) communication or other temporal mutual exclusion communication protocols among the master node **14** and the nodes **20–22**. The master node **14** and each node **20–22** may be allocated time slots the boundaries of which provide event triggers for TDMA communication. The event triggering techniques described above may be use to trigger time slot communication on the communication link **12** in each of the master node **14** and the nodes **20–22**.

The synchronization protocol **100**, with or without traceable time, may be used to provide an accurate time base for data acquisition and control in the system **10**. The techniques disclosed above for providing phase coherent signals in the master nodes **14** and nodes **20–22** enables the establishment of a periodic time base for data acquisition and control. The generation of these phase coherent signal may be used to form the basis for a class of supervisory control and data acquisition (SCADA) systems.

The techniques disclosed above for providing phase coherent signals enables the timing function to be distributed to the actual measurement or control nodes. This provides for improved accuracy, robustness and flexibility. As an example of flexibility, different portions of a SCADA system using the techniques disclosed herein may be configured to run at different periodic or scheduled frequencies without any conflict of resources. This is in contrast to prior centralized systems in which the management of two or more processes with different periodicity in general produces resource contention in the applications. This can cause timing jitter in the controlled processes.

A distributed system which generates "time ticks" using present techniques for providing phase coherent signals is more accurate than prior systems. In addition, such a system is more robust in that the synchronization protocol **100** tends to keep the local clocks in sync even in the face of lost timing data packets. Moreover, the synchronization protocol **100** consumes minimal amounts of network bandwidth in comparison to prior systems.

FIG. **3** illustrates the master clock **18** in one embodiment. The master clock **18** includes a computation engine **40**, a time-stamp latch **42**, an adjustable clock **44**, at time-stamp latch **46**, and a time packet recognizer **48**. The arrangement shown causes the time value in the adjustable clock **44** to be controlled by the traceable time source **16** and causes the time values held in the slave clocks **30–32** to be controlled by the time value in the adjustable clock **44**.

The traceable time source **16** generates an updated traceable time value **50** and a series of pulses **52** which in one embodiment occur once per second. The pulses **52** continuously cause the time-stamp latch **42** to latch time values from the adjustable clock **44**. The computation engine **40** compares the updated traceable time value **50** to the time values from the time-stamp latch **42** and issues a set of clock adjustment signals **54** that cause the adjustable clock **44** to move toward and eventually match the updated traceable time value **50**.

The computation engine **40** issues the clock adjustment signals **54** to cause the adjustable clock **44** to either speed up or slow down or maintain rate or reload with a new time value. In one embodiment, the adjustable clock **44** is implemented as a counter driven by an oscillator with an adder that adds either 0, 1, or 2 to the counter during each period of the oscillator. If the time value held in the time-stamp latch **42** is less than the updated traceable time value **50** then the computation engine **40** issues the clock adjustment signals **54** to cause a 2 to be added to the counter of the adjustable clock **44**. If the time value held in the time-stamp latch **42** equals the updated traceable time value **50** then the computation engine **40** issues the clock adjustment signals **54** to cause a 1 to be added to the counter of the adjustable clock **44**. If the time value held in the time-stamp latch **42** is greater than the updated traceable time value **50** then the computation engine **40** issues the clock adjustment signals **54** to cause a 0 to be added to the counter of the adjustable clock **44**. If the difference between the time value held in the time-stamp latch **42** and the updated traceable time value **50** is greater than a predetermined value then the computation engine **40** uses the clock adjustment signal **54** to reload the adjustable clock **44**.

The time-stamp latch **46** obtains time values from the adjustable clock **44**. The time packet recognizer **48** in response generates time data packets and transfers them via the communication link **12** to cause the slave clocks **30–32** to synchronize to the time value held in the time-stamp latch **46**. This is in accordance with the synchronization protocol **100**.

FIG. **4** illustrates a technique for introducing traceable time into a system **70** which includes a set of computer systems **60–62** that participate in a synchronization protocol

72 which differs from the synchronization protocol 100. In one embodiment, the synchronization protocol 72 implemented by the computer systems 60–62 is the network time protocol (NTP).

The computer systems 60–62 may be personal computers or workstations or a combination of these. In accordance with NTP, the computer systems 60–62 periodically exchange messages containing the local system clock time via a communication link 64. In response, each computer system 60–62 adjusts its system clock. Eventually, the system clocks in the computer systems 60–62 converge.

The computer system 60 not only implements the NTP protocol, but also includes a slave clock 66 which synchronizes to the traceable time values distributed by the master clock 18 using the synchronization protocol 100. The software layers of the NTP protocol in the computer system 60 obtain time values from the slave clock 66 rather than the system clock of the computer system 60. In addition, the slave clock 66 remains synchronized to the master clock 18 rather than converging with the NTP protocol as do the system clocks of the computer systems 61–62. As a result, the system clocks of the computer systems 61–62 eventually converge to the time value in the slave clock 66 which is traceable time.

The slave clock 66 may be implemented on a standardized interface card for the computer system 60. The same interface card may contain both the slave clock 66 and the elements needed for communication on the communication link 64. For example, one interface card may contain the slave clock 66 and an Ethernet interface if the communication link 64 is Ethernet.

In one embodiment, the master node 14 is a computer system such as a personal computer or workstation. In such an embodiment, the host processor of the master node 14 may perform the functions associated with the computation engine 40.

FIG. 5 illustrates a technique for using the synchronization protocol 100 to distribute the time values obtained with the synchronization protocol 72. A system 80 is shown including a set of computer systems 90–92 that participate in the synchronization protocol 72 which may be NTP via a communication link 82. The computer system 90 includes a master clock 86 which participates in the synchronization protocol 100 via the communication link 82 with a node 84 having a slave clock 88. The computer system 90 obtains a time value from its system clock and provides it to the master clock 86. The master clock 86 distributes this time value to the slave clock 88 using the synchronization protocol 100.

The master clock 86 may be implemented on a standardized interface card for the computer system 90. The same interface card may contain both the master clock 86 and the elements needed for communication on the communication link 82. The host processor of the computer system 90 may perform the functions associated with the computation engine of the master clock 86. Alternatively, the master clock 86 may contain a computation engine which obtains system time values from the host processor of the computer system 90.

FIG. 6 shows a technique for distributing traceable time values to nodes which are connected to different sub-nets of a system 150. One sub-net of the system 150 includes a set of nodes 120–122 coupled to a communication link 110 and another sub-net of the system 150 includes a set of nodes 130–132 coupled to a communication link 112.

Traceable time values are introduced into the system 150 using a master node 140 which includes a traceable time source 144 and a master clock 146. The traceable time source 144 provides traceable time values to the master clock 146 and the master clock 146 distributes the traceable time values to a set of slave clocks 140–142 in the nodes 120–122 using the synchronization protocol 100.

The system 150 includes a boundary node 142 coupled between the communication links 110–112. The boundary node 142 includes a master/slave clock 152. The master/slave clock 152 behaves like a slave clock in that it synchronizes to traceable time values distributed by the master clock 146 via the communication link 110. In addition, the master/slave clock 152 behaves like a master clock in that it in turn distributes the traceable time values to a set of slave clocks 150–152 in the nodes 130–132 via the communication link 112.

FIG. 7 illustrates one embodiment of the master/slave clock 152. The master/slave clock 152 includes an adjustable clock 254 along with a time packet recognizer 250 and a latch 252 for the slave side on the communication link 110 and a time packet recognizer 258 and a latch 256 for the master side on the communication link 112. A computation engine 260 performs the computations for both the master and slave sides of the master/slave clock 152.

The operation of the slave side of the master/slave clock 152 is as follows. The master clock 146 transfers a timing data packet (a timing event) via the communication link 110 and then transfers a data packet containing a traceable time value for the timing event (the supplemental information for the timing event) via the communication link 110. In response to the timing event, the time packet recognizer 250 causes the latch 252 to latch the local time in the adjustable clock 254. The computation engine 260 receives the supplemental information and computes the difference between the time value in the latch 252 and the traceable time value from the supplemental information and then adjusts the adjustable clock 254 accordingly. This causes the adjustable clock 254 to sync to the traceable time values distributed by the master clock 146.

The operation of the master side of the master/slave clock 152 is as follows. The computation engine 260 periodically generates a timing data packet and transfers it via the communication link 112 to the slave clocks 150–152. In response to a timing data packet, the time packet recognizer 258 causes the latch 256 to latch the local time in the adjustable clock 254. The computation engine 260 then obtains the latched time value from the latch 256 and transfers it as supplemental information via the communication link 112 to the slave clocks 150–152. This enables the computation engines in the slave clocks 150–152 to sync up to traceable time values of the adjustable clock 254. For example, the slave clock 150 uses the difference between its reception time of a timing data packet and the traceable time value provided by the corresponding supplemental information to adjust its local adjustable clock.

The computation engine 260 may render a determination of which is the master side and which is the slave side of the master/slave clock 152 in response to information provided by the clocks attached to each communication link 110–112. This information may include the accuracy of the clocks, whether a clock provides traceable time values either directly or as a slave to a traceable time source, and whether clocks are themselves two-sided clocks of a boundary node.

A master/slave clock in a boundary node in general has one slave side and n master sides for linking up to n additional subnets. Each slave side and each of the n master sides have corresponding time packet recognizers and

latches and the same computation engine may be shared among them. The slave side is a slave on the subnet which has the best source of time according to a preselected criteria such as accuracy or any of the other criteria set forth above.

FIG. 8 shows a data acquisition and control system **200** which includes a set of cards **210–213** connected to a backplane **202**. The cards **210–213** each include circuitry for providing synchronized clocks using the synchronization protocol **100** and include circuitry for generating phase coherent signals using techniques outline above.

The synchronization protocol **100** requires that two basic types of information be communicated among the cards **210–213**. The basic types are a timing event and a set of supplementary information associated with the timing event. For example, a timing data packet is a timing event. If traceable time is provided by a GPS source then the timing data packets usually occur at one per second.

The supplemental information for a timing event is an interpretation of the timing event. For example, if a timing event is associated with a GPS source then the supplemental information for the timing event is the UTC time at which the timing event was generated. If a timing event is associated with a master clock then the supplemental information for the timing event is the time at which the node having the master clock generated the timing event.

The cards **210–213** communicate timing events using a set of communication links **220–223** arranged as a daisy chain. Each of the cards **210–213** includes a daisy chain input port (IN) and a daisy chain output port (OUT). Each of the cards **210–213** receives timing events on its IN port and issues timing events on its OUT port. If a card does not detect any timing events on its IN port then it can assume that it is the master of the data acquisition and control system **200**. Alternatively, the master of the data acquisition and control system **200** may be configured by a host processor connected to the backplane **202**. The one of the cards **210–213** that is master issues timing events at an appropriate rate. Each of the remaining ones of the cards **210–213** that is a slave card passes each timing event from its IN port to its OUT port without significant delay.

In one embodiment, the communication links **220–223** are signal lines. In another embodiment, the communication links **220–223** are fiber optic links. The cards **210–223** may include LEDs and photo-diodes so that when each card **210–223** is inserted in the backplane **202** a daisy chain input and output link is formed. In other embodiments, the communication links **220–223** may be RS232 links or network communication links such as Ethernet.

The one of the cards **210–213** having the master clock generates the supplemental information and may be referred to as the master card. The remainder of the cards **210–213** receive the supplemental information and may be referred to as slave cards. In one embodiment, the cards **210–213** communicate the supplemental information using communication paths on the backplane **202** since the supplemental information is less time critical than the timing events and therefore the delays associated with communication on the backplane **202** may be tolerable. This embodiment allows simple and low cost implementations of the communication links **220–223**. For example, the communication links **220–223** may be signal lines that carry CMOS signals which communication timing events.

In an alternative embodiment, the cards **210–213** communicate the supplemental information using the communication links **220–223**. This embodiment may require that the communication links **220–223** be implemented in a somewhat more complex manner. For example, information may be carried on the communication links **220–223** using a signal packet. A start of frame portion of the signal packet may indicate the timing event and the supplemental information may be encoded in the remainder of the signal packet. Another option is an RS232 implementation of the communication links **220–223**.

In one embodiment, the card **213** is the master card which includes a master clock that distributes time values to slave clocks in the cards **210–212** and to slave clocks in nodes that are connected to a communication link **204**. These time values may be obtained from a host processor connected to the backplane **202** which implements the NTP protocol.

In another embodiment, the card **213** includes a master clock that distributes time values to slave clocks in the cards **210–212** and includes a slave clock that synchronizes to a master clock in a node connected to the communication link **204**. The node connected to the communication link **204** and having the master clock may include a traceable time source such as GPS as described above. The communication link **204** may be a network communication link **204** such as Ethernet. The card **213** may also serve as a port to the communication link **204** for a host processor connected to the backplane **202**.

The computations needed for the synchronization protocol **100** may be performed by computation circuitry implemented on the cards **210–213**. Alternatively, the computations needed for the synchronization protocol **100** may be performed by a host processor connected to the backplane **202**.

The backplane **202** may be that of a personal computer or workstation or a specialized system. The backplane **202** may be a standard backplane such as PCI, VME, or ISA to name a few examples.

A communication link for communicating the timing events and the supplemental information for the synchronization protocol **100** may be implemented as an additional trace on the backplane **202**. However, this may require a modification to an existing backplane standard which may not be practical. In addition the communication link for communicating the timing events and the supplemental information may be a packetized network such as Ethernet which is connected to each of the cards **210–213**. This may be too expensive to implement for a backplane system.

The foregoing detailed description of the present invention is provided for the purposes of illustration and is not intended to be exhaustive or to limit the invention to the precise embodiment disclosed. Accordingly, the scope of the present invention is defined by the appended claims.

What is claimed is:

1. A system, comprising:

a set of nodes coupled to a communication link and each having a slave clock that synchronizes a slave time value using a synchronization protocol;

traceable time source that generates a traceable time value;

master node coupled to the communication link and having a master clock that synchronizes a master time value to the traceable time value and that distributes the master time value to the slave clocks via the communication link using the synchronization protocol.

2. The system of claim **1**, wherein the traceable time source is contained in the master node.

3. The system of claim **1**, wherein the traceable time source is a GPS receiver.

4. The system of claim **1**, further comprising a computer system coupled to the communication link and having a

slave clock which synchronizes to the master time value via the communication link using the synchronization protocol, the computer system executing a set of software for synchronizing a system clock of the computer system to a set of other system clocks in a set of other computer systems coupled to the communication link such that the computer system substitutes a time value from the slave clock for a time value in the system clock for use in synchronizing the other system clocks in the other computer systems.

5. The system of claim 4, wherein the communication link enables communication using a packetized network communication protocol.

6. The system of claim 5, wherein the slave clock of the computer system and circuitry for the packetized network communication protocol are both implemented on an interface card for the computer system.

7. A system, comprising:

a set of nodes coupled to a communication link and each having a slave clock that synchronizes a slave time value using a synchronization protocol;

computer system coupled to the communication link and executing a set of software for synchronizing a system clock of the computer system to a set of other system clocks in a set of other computer systems coupled to the communication link, the computer system having a master clock that synchronizes a master time value to the system clock and that distributes the master time value to the slave clocks via the communication link using the synchronization protocol.

8. The system of claim 7, wherein the communication link enables communication using a packetized network communication protocol.

9. The system of claim 8, wherein the master clock and circuitry for the packetized network communication protocol are implemented on an interface card for the computer system.

10. A system, comprising:

first sub-net having a set of first nodes coupled to a first communication link, each first node having a slave clock that synchronizes a slave time value using a synchronization protocol on the first communication link;

second sub-net having a set of second nodes coupled to a second communication link, each second node having a slave clock that synchronizes a slave time value using the synchronization protocol on the second communication link;

traceable time source that generates a traceable time value;

master node coupled to the first communication link and having a master clock that synchronizes a master time value to the traceable time value and that distributes the master time value to the slave clocks via the first communication link using the synchronization protocol;

boundary node coupled to the first communication link and having a master/slave clock that synchronizes to the master time value distributed by the master clock using the synchronization protocol on the first communication link and that distributes the master time value to the slave clocks of the second sub-net via the second communication link using the synchronization protocol.

11. The system of claim 10, wherein the traceable time source is contained in the master node.

12. The system of claim 10, wherein the traceable time source is a GPS receiver.

13. A system, comprising:

master node coupled to a communication link and having a master clock that distributes a master time value via the communication link using a synchronization protocol;

a set of nodes coupled to the communication link and each having a slave clock that synchronizes a slave time value to the master time value using the synchronization protocol, wherein one or more of the nodes includes circuitry for generating a phase coherent periodic signal in response to the slave time value.

14. The system of claim 13, wherein the phase coherent periodic signal is used for synchronizing TDMA communication on the communication link.

15. The system of claim 13, wherein the phase coherent periodic signal is used for a time base in data acquisition and control.

16. The system of claim 13, further comprising a traceable time source that generates a traceable time value such that the master clock synchronizes the master time value to the traceable time value.

17. A data acquisition and control system, comprising:

a set of slave cards connected to a backplane, each slave card having a slave clock for use with a synchronization protocol which includes at least one timing event and a set of supplemental information associated with the timing event;

master card connected to the backplane and having a master clock for use with the synchronization protocol;

daisy chain communication link among the master card and the slave cards such that the master card communicates the timing event to the slave cards using the daisy chain communication link and communicates the supplemental information to the slave cards using a communication path on the backplane.

18. The data acquisition and control system of claim 17, wherein the master card is coupled to a network communication link which is connected to one or more nodes each of which includes a slave clock such that the master clock in the master card distributes the timing event and the supplemental information to the nodes via the network communication link.

* * * * *

US006278718B1

(12) **United States Patent**
Eschholz

(10) **Patent No.: US 6,278,718 B1**
(45) **Date of Patent: *Aug. 21, 2001**

(54) **DISTRIBUTED NETWORK SYNCHRONIZATION SYSTEM**

(75) Inventor: **Siegmar K. Eschholz**, Bourne, MA (US)

(73) Assignee: **Excel, Inc.**, Hyannis, MA (US)

(*) Notice: This patent issued on a continued prosecution application filed under 37 CFR 1.53(d), and is subject to the twenty year patent term provisions of 35 U.S.C. 154(a)(2).

Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

(21) Appl. No.: **08/705,226**

(22) Filed: **Aug. 29, 1996**

(51) **Int. Cl.**[7] **H04J 3/06**
(52) **U.S. Cl.** **370/503**; 370/509
(58) **Field of Search** 370/503, 507, 370/508, 512, 517, 389, 506, 509, 510

(56) **References Cited**

U.S. PATENT DOCUMENTS

Re. 31,852	3/1985	Soderblom	370/85.15
4,038,638	7/1977	Hwang	379/272
4,173,713	11/1979	Giesken et al.	370/65
4,228,536	10/1980	Gueldenpfennig	370/66
4,229,816	10/1980	Breidenstein et al.	370/100.1
4,456,987	6/1984	Wirsing	370/65.5
4,527,012	7/1985	Caplan et al.	379/284
4,539,676	9/1985	Lucas	370/60
4,569,041	2/1986	Takeuchi et al.	370/85.12
4,686,330	8/1987	Hourton	379/269
4,757,497	7/1988	Beierle et al.	370/85.12
4,805,172	2/1989	Barbe et al.	370/68.1
4,962,497	10/1990	Ferenc et al.	379/89
5,029,199	7/1991	Jones et al.	379/89
5,107,490 *	4/1992	Wilson	370/404
5,119,370	6/1992	Terry	370/60.1

(List continued on next page.)

FOREIGN PATENT DOCUMENTS

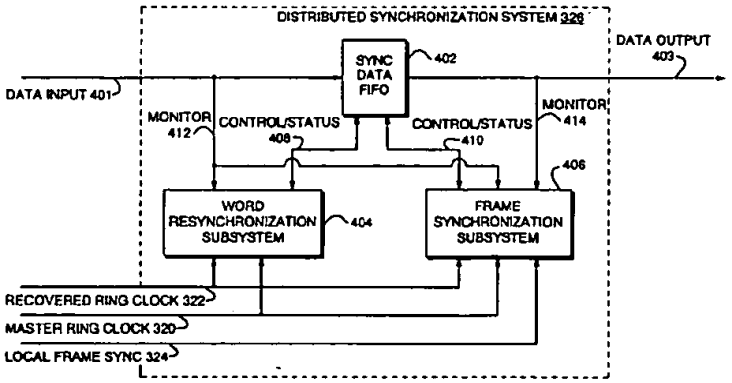
0119105	9/1984	(EP) .	
0256526	2/1988	(EP) .	
0472380	2/1992	(EP) .	
2538662	12/1982	(FR) .	
2 693 333	7/1994	(FR)	H04J/3/22
1243464	8/1971	(GB) .	
86/04203	7/1986	(WO)	H04L/11/16
9416528	7/1994	(WO) .	

Primary Examiner—Douglas Olms
Assistant Examiner—Ricardo Pizarro
(74) *Attorney, Agent, or Firm*—Cesari and McKenna, LLP

(57) **ABSTRACT**

A distributed synchronization system for use in each node of a distributed asynchronous telecommunications network system that continually monitors and controls the flow of data through an implementing node to prevent dataflow errors due to phase and frequency differences in source and destination nodal clocks, and to control inter-nodal network latency so as to support the transmission of synchronous data. A synchronization data FIFO buffers predetermined fields or portions of fields of a unique frame packet received from a source node before retransmission to a destination node on the network. The frame packet includes a frame synchronization field indicating the beginning of a new frame packet; a payload field containing valid data; and a dead zone field providing bandwidth during which the present invention performs synchronization functions. A frame synchronization subsystem, implemented in a designated master node, guarantees that a frame is released at the beginning of an independently-determined frame regardless of network latency. A word resynchronization subsystem manages the flow of data through the data FIFO of each non-master node, receiving and storing data at the source node's clock rate and transmitting the data according to its own clock, thereby guaranteeing the efficient receipt and transmission of data between asynchronously-communicating nodes.

6 Claims, 11 Drawing Sheets



U.S. PATENT DOCUMENTS

5,241,543		8/1993	Amada et al.	370/100.1
5,282,200	*	1/1994	Dempsey	370/522
5,327,425	*	7/1994	Niwa	370/505
5,349,579		9/1994	Madonna et al.	370/58.2
5,452,305	*	9/1995	Nagatake	370/516
5,473,610	*	12/1995	Rainard	370/518
5,517,489	*	5/1996	Ogura	370/223
5,517,505	*	5/1996	Buchholz	370/508
5,544,163	*	8/1996	Madonna	370/352
5,557,609	*	9/1996	Shobatake	370/509
5,742,600	*	4/1998	Nishihara	370/395
5,778,188	*	7/1998	Taniguchi	395/200.66

* cited by examiner

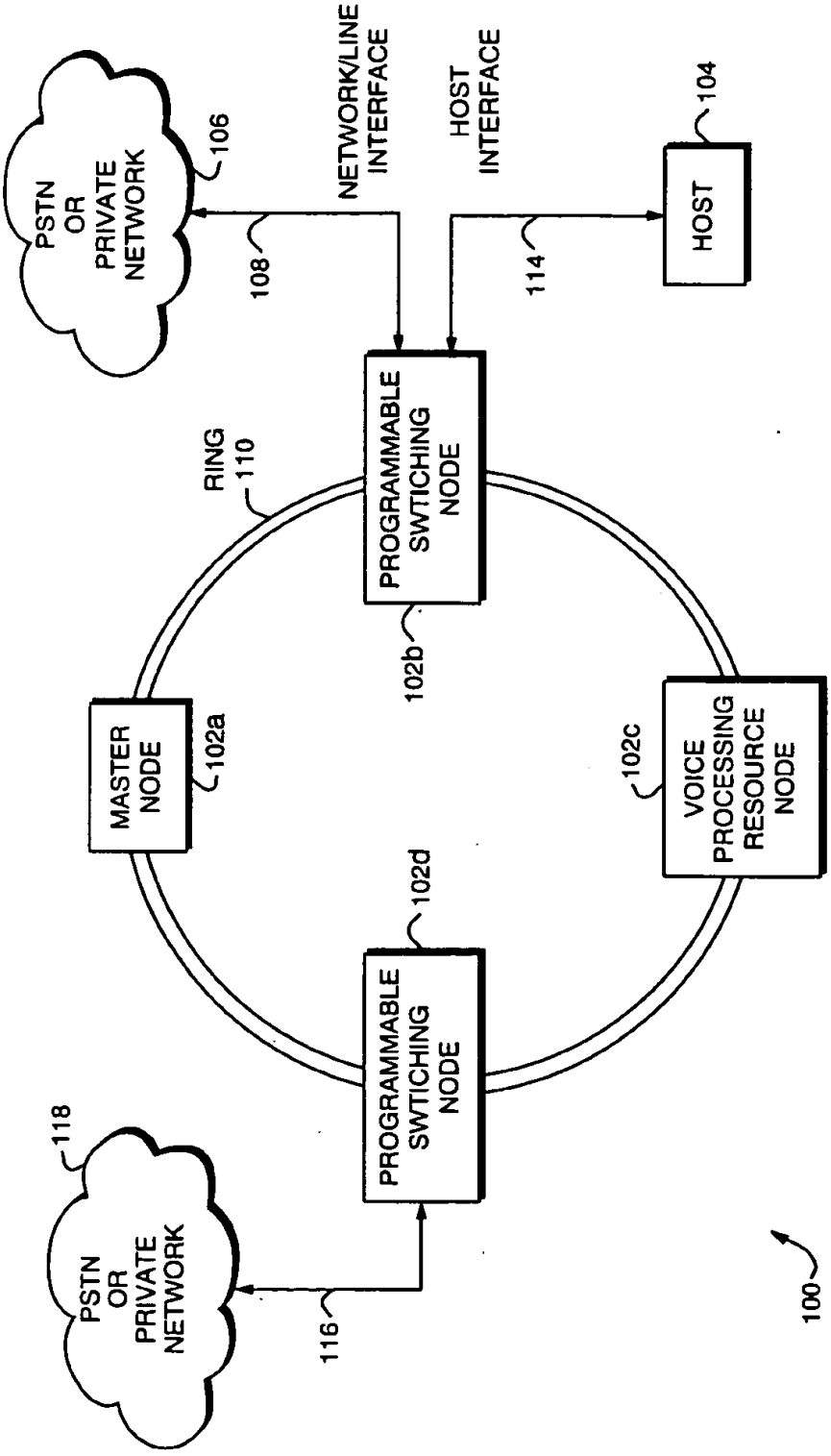

PSTN OR PRIVATE NETWORK
106
NETWORK/LINE INTERFACE
HOST INTERFACE
104
HOST
108
114
RING
110
PROGRAMMABLE SWTICHING NODE
102b
MASTER NODE
102a
VOICE PROCESSING RESOURCE NODE
102c
102d
PROGRAMMABLE SWTICHING NODE
118
PSTN OR PRIVATE NETWORK
116
100

FIG. 1

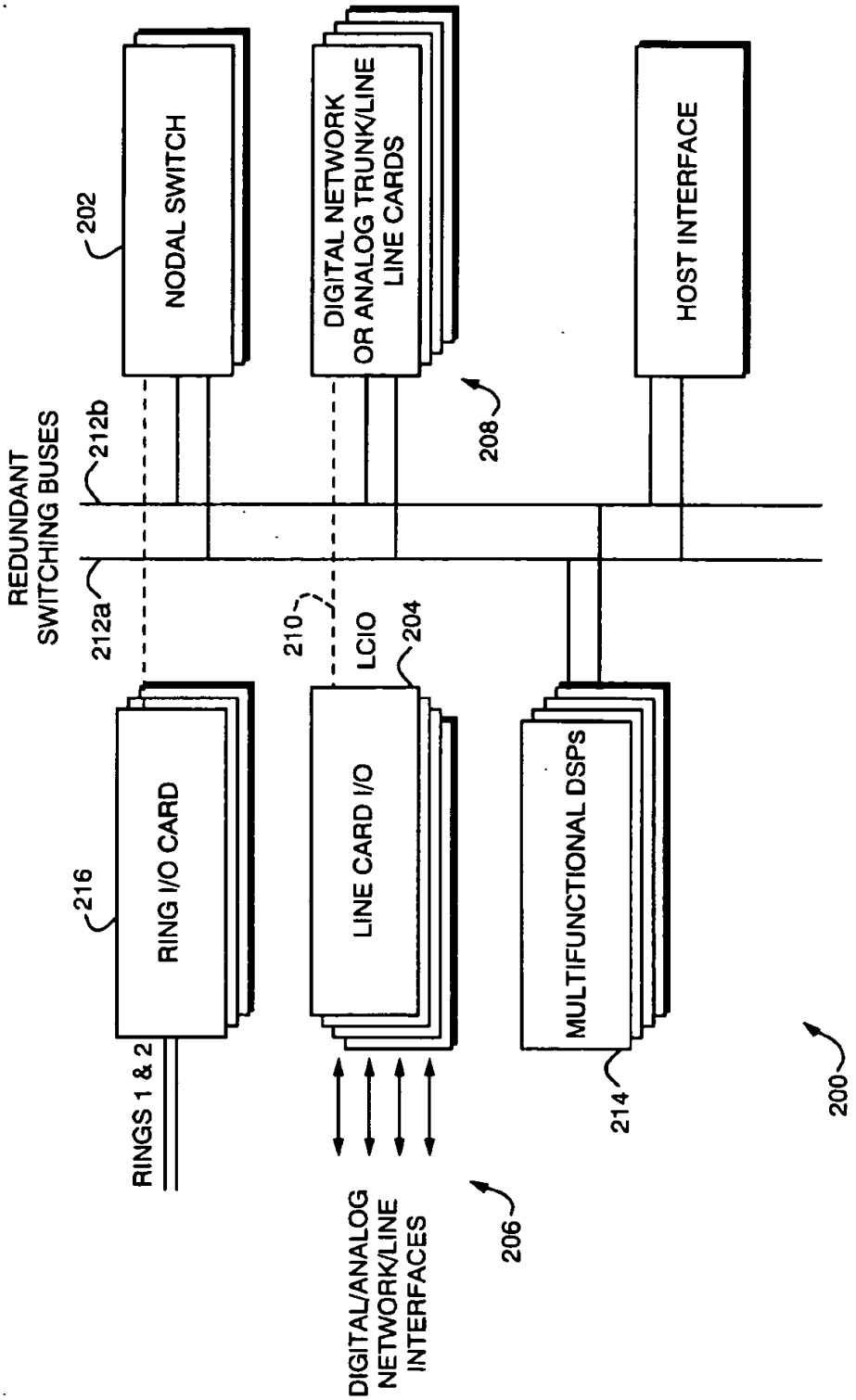

NODAL SWITCH
202
DIGITAL NETWORK OR ANALOG TRUNK/LINE LINE CARDS
HOST INTERFACE
208
REDUNDANT SWITCHING BUSES
212b
212a
210
LCIO
204
RING I/O CARD
216
LINE CARD I/O
MULTIFUNCTIONAL DSPs
214
200
RINGS 1 & 2
DIGITAL/ANALOG NETWORK/LINE INTERFACES
206

FIG. 2

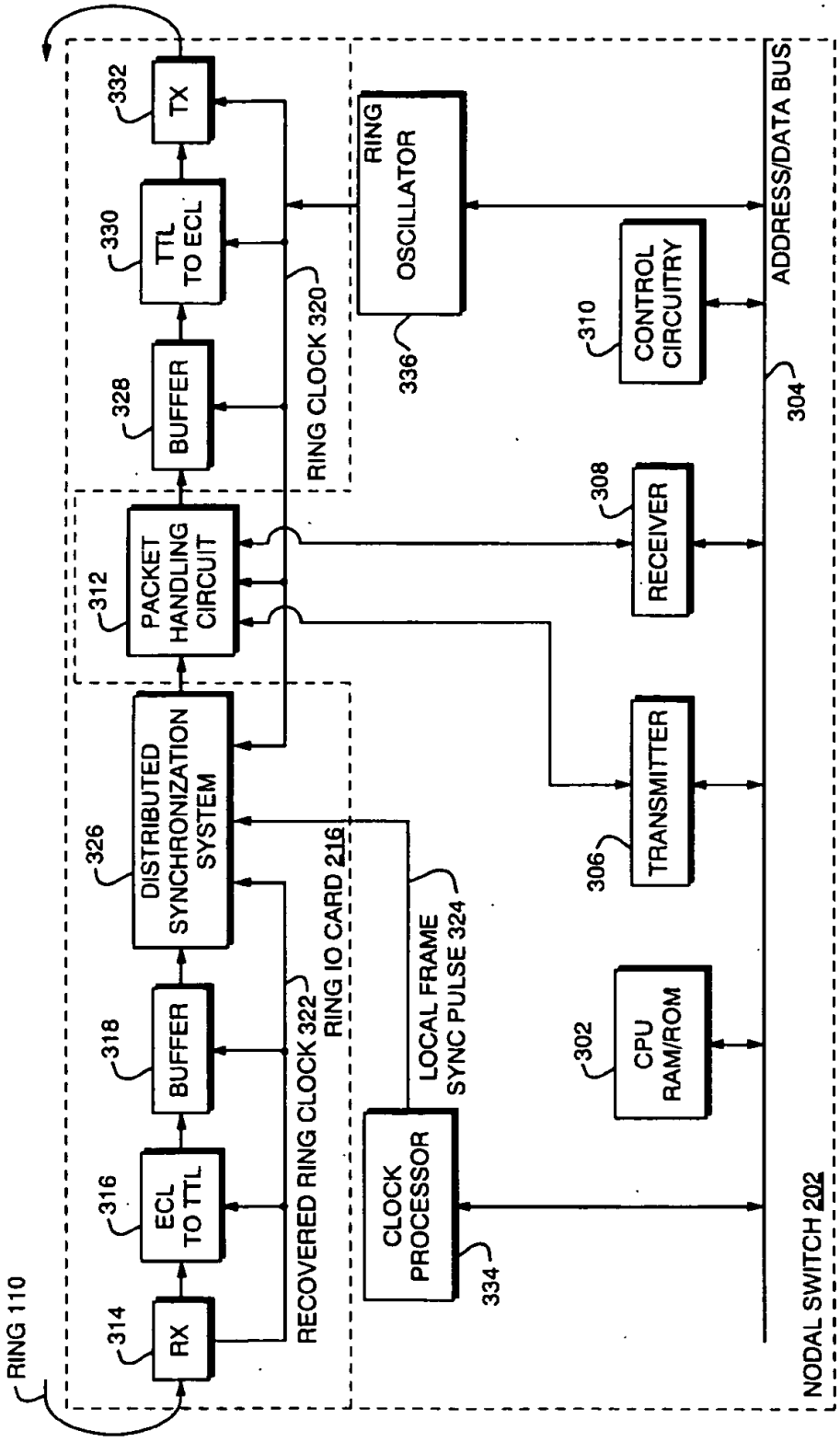

RING 110
314
RX
316
ECL TO TTL
318
BUFFER
326
DISTRIBUTED SYNCHRONIZATION SYSTEM
312
PACKET HANDLING CIRCUIT
328
BUFFER
330
TTL TO ECL
332
TX
RECOVERED RING CLOCK 322
RING IO CARD 216
RING CLOCK 320
LOCAL FRAME SYNC PULSE 324
CLOCK PROCESSOR
334
RING OSCILLATOR
336
302
CPU RAM/ROM
306
TRANSMITTER
308
RECEIVER
310
CONTROL CIRCUITRY
304
ADDRESS/DATA BUS
NODAL SWITCH 202

FIG. 3

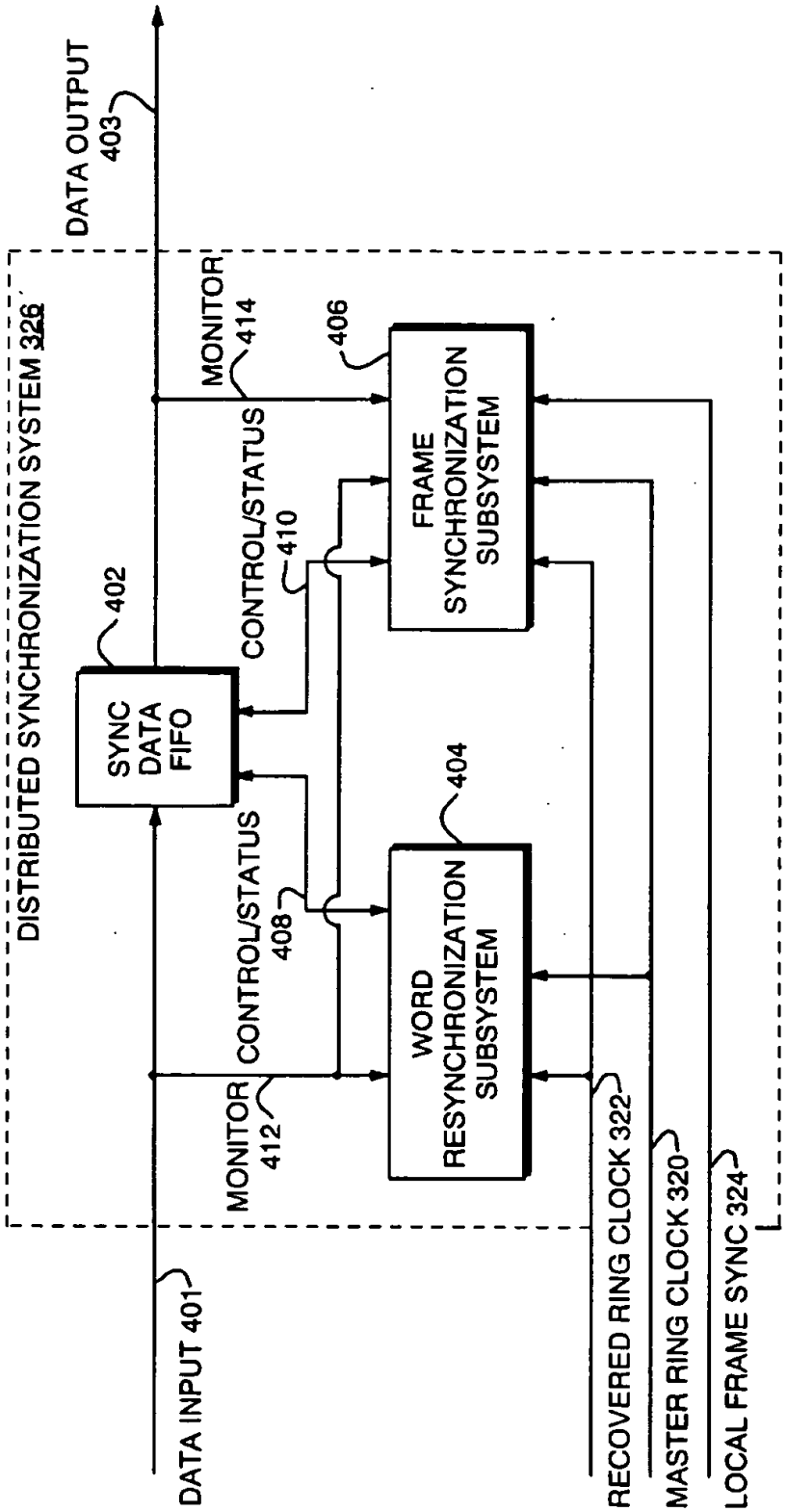

DISTRIBUTED SYNCHRONIZATION SYSTEM 326
DATA OUTPUT 403
MONITOR 414
406
FRAME SYNCHRONIZATION SUBSYSTEM
CONTROL/STATUS 410
402
SYNC DATA FIFO
CONTROL/STATUS 408
404
WORD RESYNCHRONIZATION SUBSYSTEM
MONITOR 412
DATA INPUT 401
RECOVERED RING CLOCK 322
MASTER RING CLOCK 320
LOCAL FRAME SYNC 324

FIG. 4

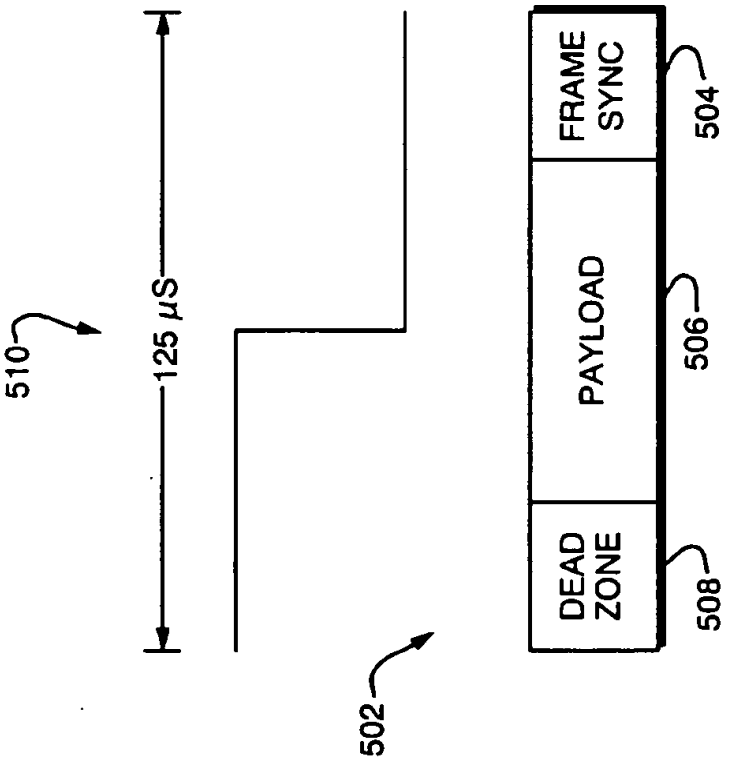


FIG. 5

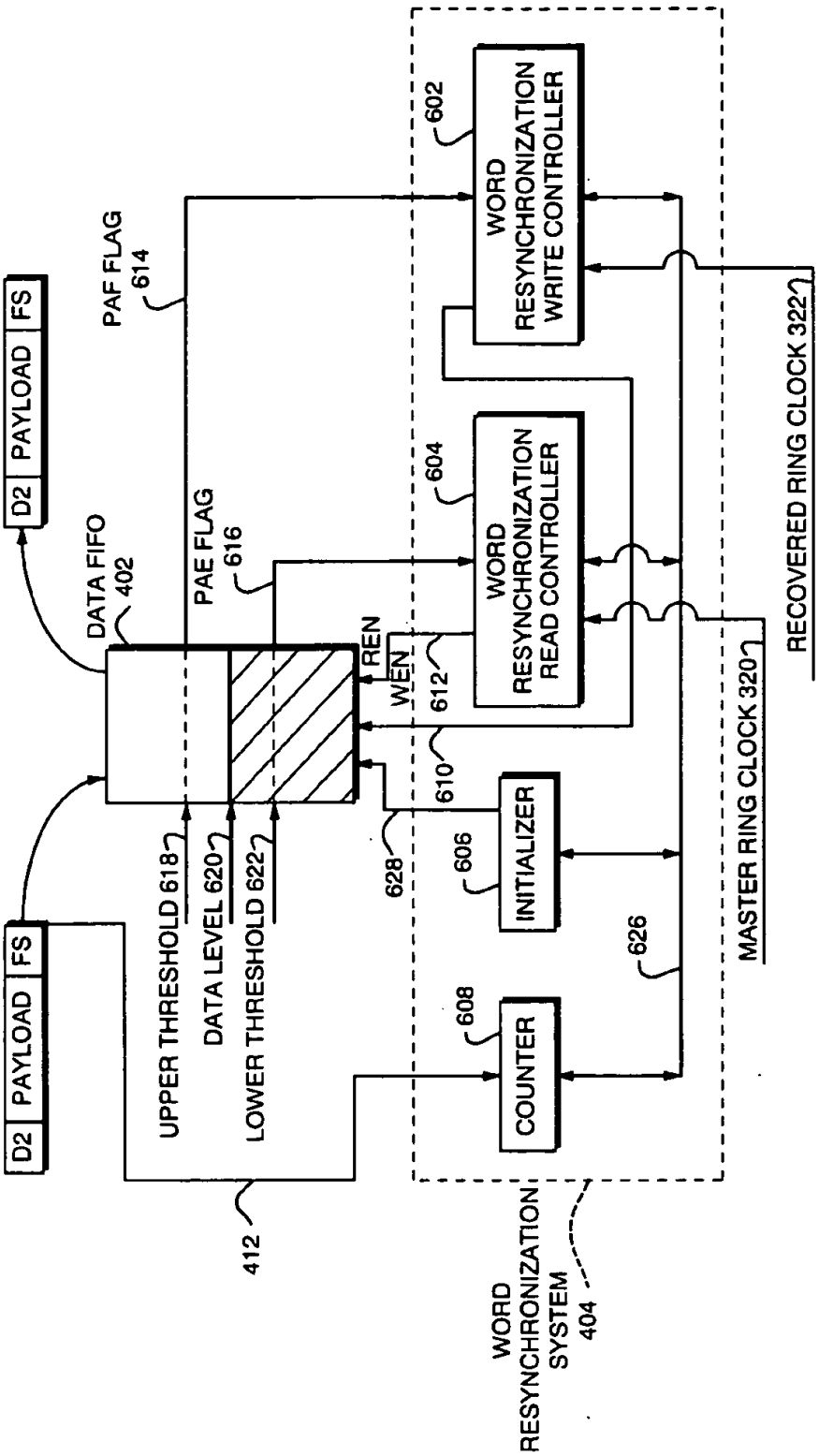

D2
PAYLOAD
FS
DATA FIFO
402
PAF FLAG
614
PAE FLAG
616
UPPER THRESHOLD 618
DATA LEVEL 620
LOWER THRESHOLD 622
412
REN
WEN
612
610
628
602
WORD RESYNCHRONIZATION WRITE CONTROLLER
604
WORD RESYNCHRONIZATION READ CONTROLLER
606
INITIALIZER
608
COUNTER
626
RECOVERED RING CLOCK 322
MASTER RING CLOCK 320
WORD RESYNCHRONIZATION SYSTEM 404

FIG. 6

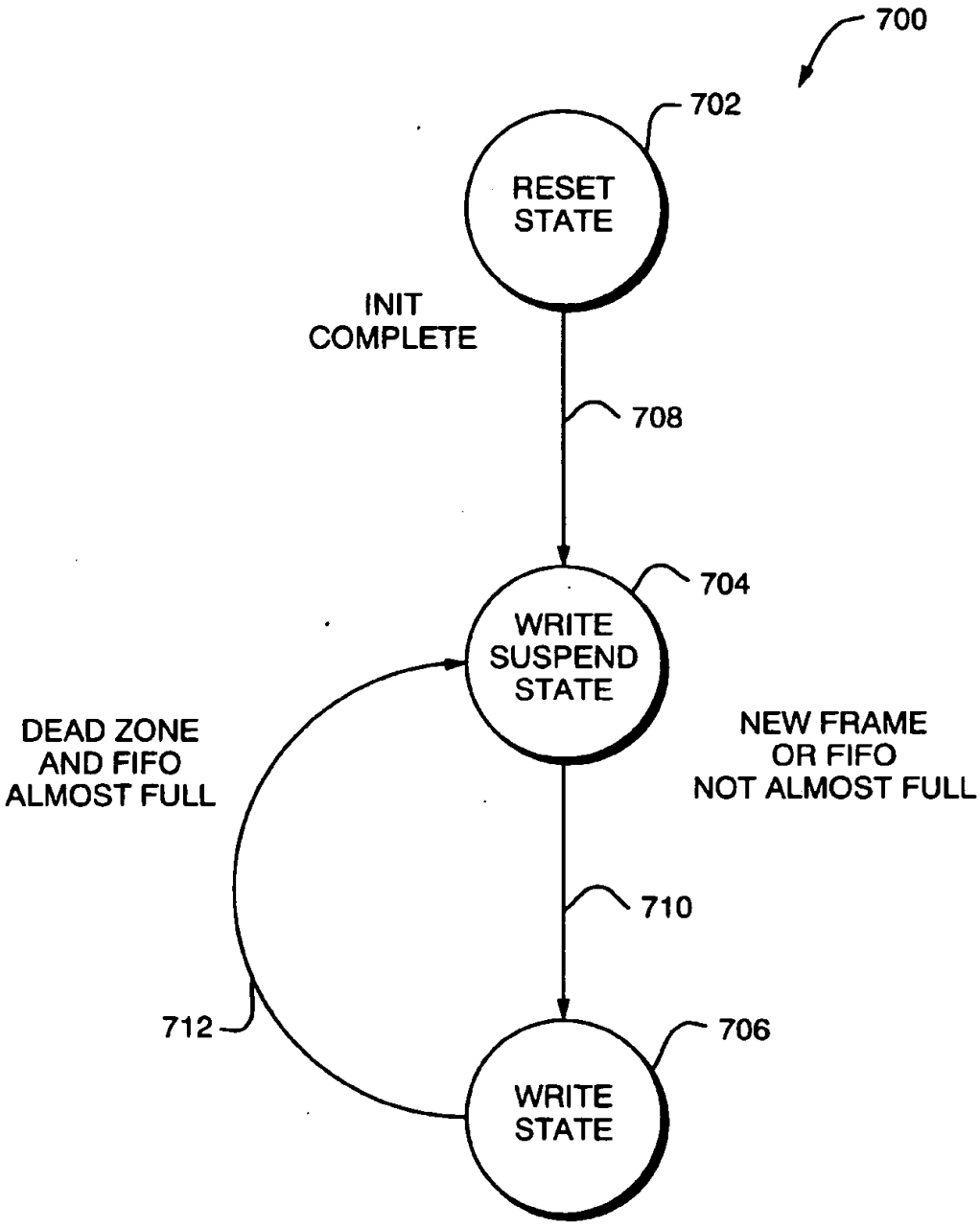

700
702
RESET STATE
INIT COMPLETE
708
704
WRITE SUSPEND STATE
DEAD ZONE AND FIFO ALMOST FULL
NEW FRAME OR FIFO NOT ALMOST FULL
710
712
706
WRITE STATE

FIG. 7

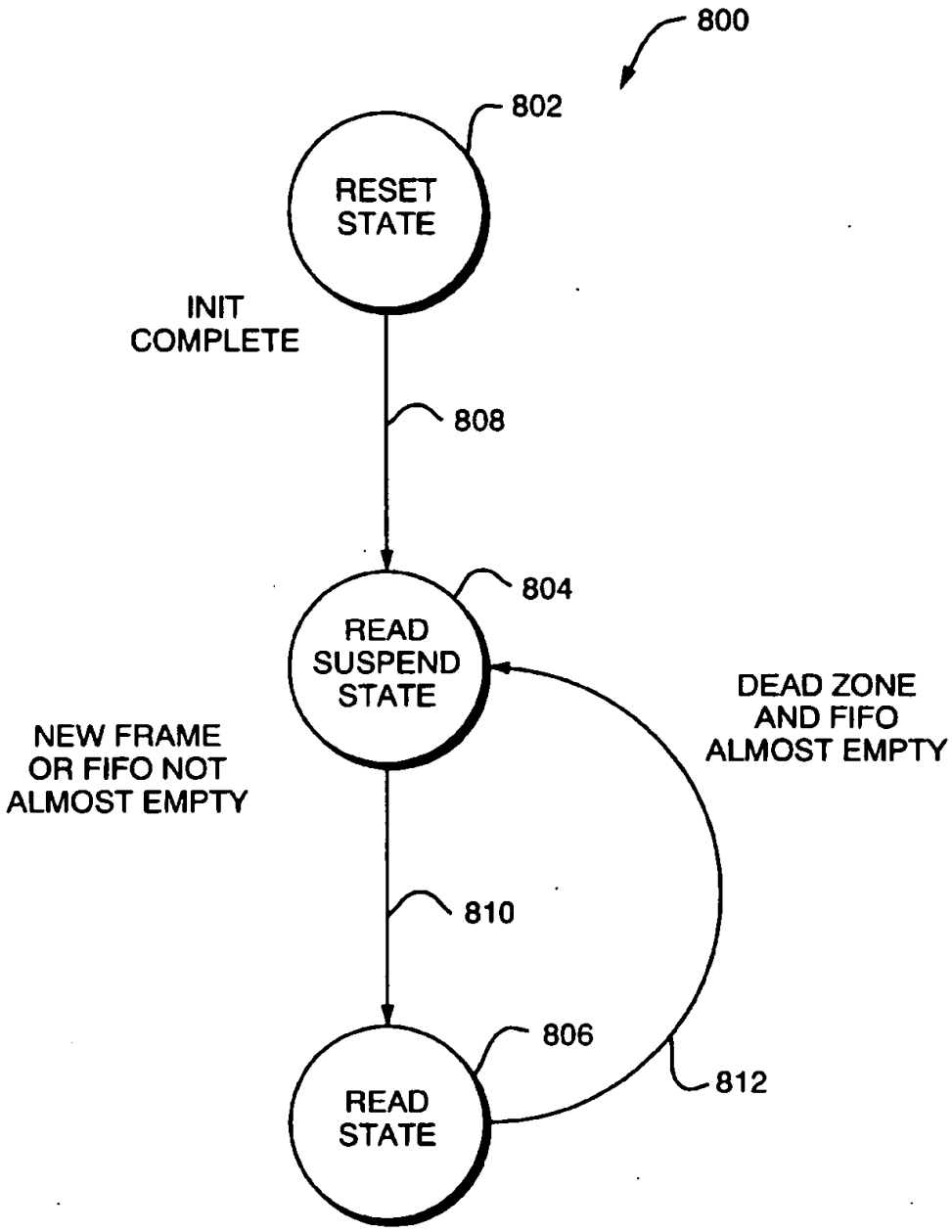

800
802
RESET STATE
INIT COMPLETE
808
804
READ SUSPEND STATE
DEAD ZONE AND FIFO ALMOST EMPTY
NEW FRAME OR FIFO NOT ALMOST EMPTY
810
806
READ STATE
812

FIG. 8

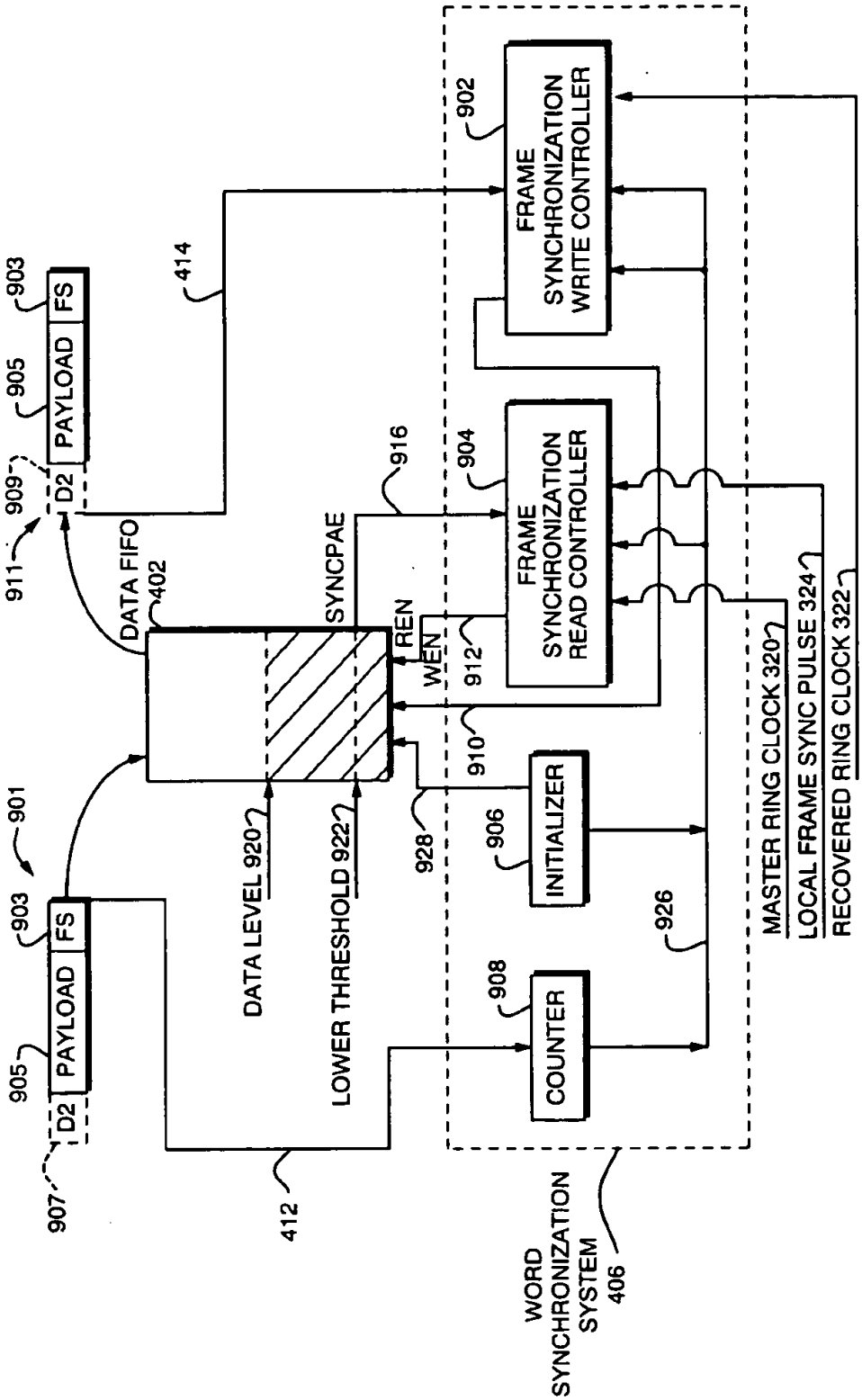

FRAME SYNCHRONIZATION WRITE CONTROLLER
902
FRAME SYNCHRONIZATION READ CONTROLLER
904
INITIALIZER
906
COUNTER
908
414
903
FS
905
PAYLOAD
D2
909
911
DATA FIFO
402
SYNCPAE
916
REN
WEN
912
910
928
901
907
412
DATA LEVEL 920
LOWER THRESHOLD 922
926
MASTER RING CLOCK 320
LOCAL FRAME SYNC PULSE 324
RECOVERED RING CLOCK 322
WORD SYNCHRONIZATION SYSTEM 406

FIG. 9

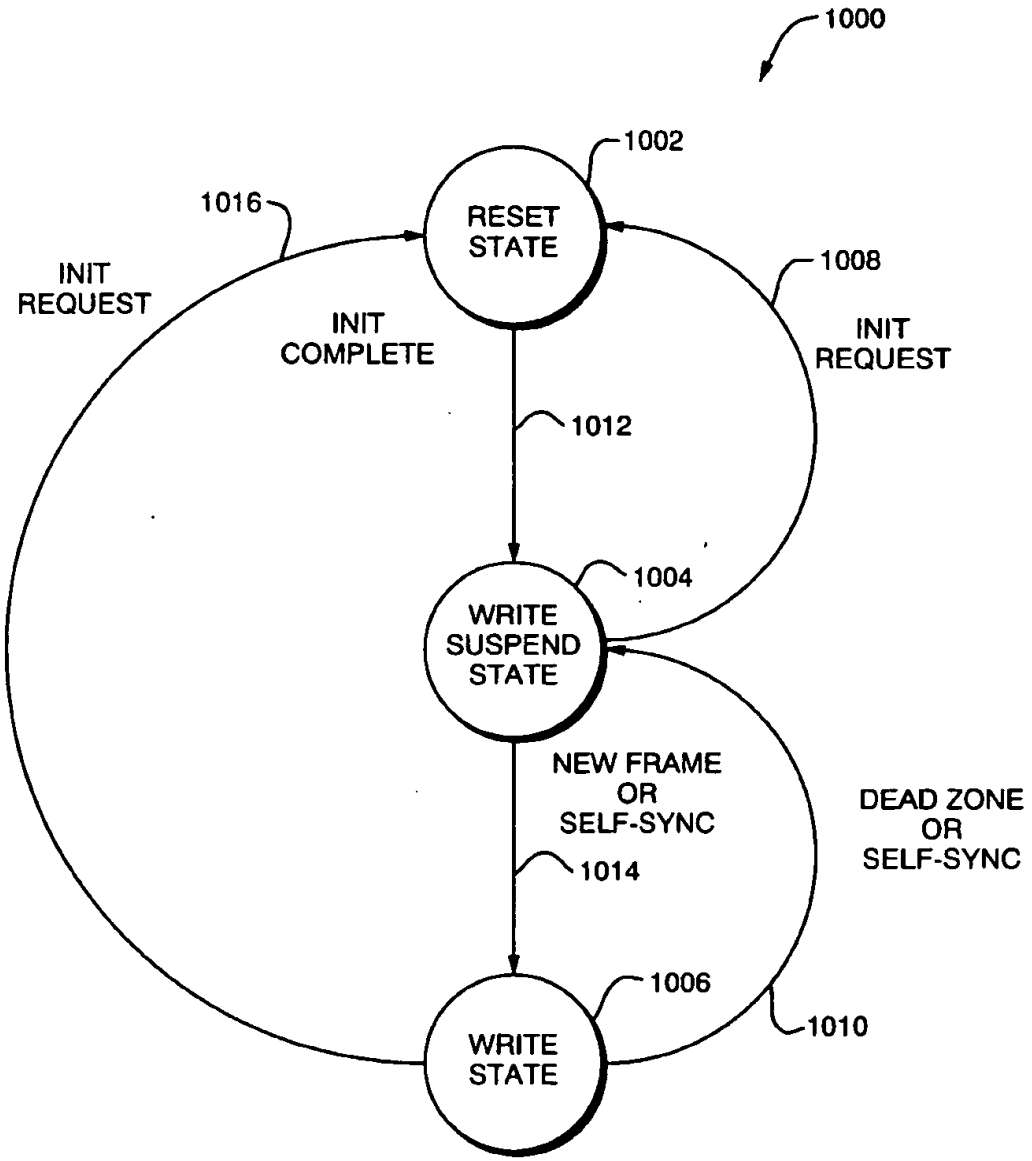

1000
1002
RESET STATE
1016
INIT REQUEST
INIT COMPLETE
1008
INIT REQUEST
1012
1004
WRITE SUSPEND STATE
NEW FRAME OR SELF-SYNC
1014
DEAD ZONE OR SELF-SYNC
1006
1010
WRITE STATE

FIG. 10

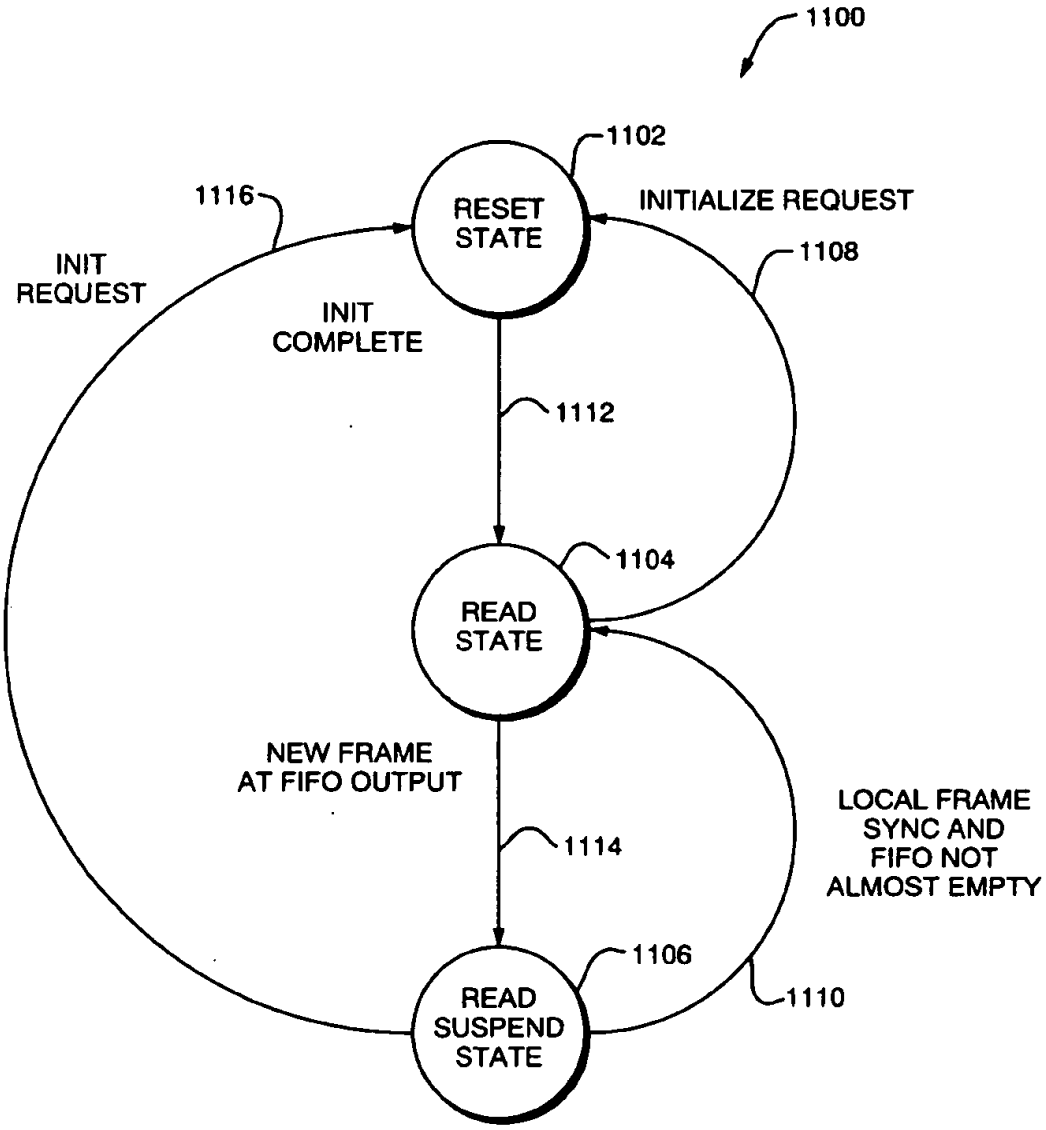

1100
1102
RESET STATE
INITIALIZE REQUEST
1108
1116
INIT REQUEST
INIT COMPLETE
1112
1104
READ STATE
NEW FRAME AT FIFO OUTPUT
1114
LOCAL FRAME SYNC AND FIFO NOT ALMOST EMPTY
1106
1110
READ SUSPEND STATE

FIG. 11

DISTRIBUTED NETWORK SYNCHRONIZATION SYSTEM

CROSS-REFERENCES TO RELATED APPLICATIONS AND PATENTS

The following commonly-owned patents and applications are related to the present invention and are incorporated by reference by this and other references:

- U.S. Pat. No. 5,544,163, entitled "Expandable Telecommunications System," naming as inventor Robert P. Madonna;
- U.S. Pat. No. 5,426,694, entitled "Telecommunications Switch Having Programmable Network Protocols and Communications Services," naming as inventor Mark P. Hebert;
- U.S. patent application Ser. No. 08/566,414, entitled "Telecommunications Switch Having A Universal Applications Program Interface," naming as inventor Mark P. Hebert, filed on Nov. 30, 1995; and
- U.S. Pat. No. 5,349,579, entitled "Telecommunications Switch With Programmable Communications Services," naming as inventors Robert P. Madonna and Kevin C. Kicklighter.

BACKGROUND OF THE INVENTION

1. Field of the Invention

This present invention relates generally to the field of telecommunications and, more specifically, to a distributed synchronization system for maintaining word and frame synchronization among asynchronous nodes of a distributed telecommunications system.

2. Related Art

Generally, a distributed network system includes a physical carrier transport system that transports data between geographically-distributed nodes. The network may take on any one of a number of architectural forms such as a bus or ring. In a bus topology, a coaxial cable or an optical fiber is commonly used as the physical carrier. Ethernet, for example, uses a coaxial cable as its transmission medium. In the case of a ring, the physical medium may be a twisted-wire pair, coaxial cable, or optical fiber. In a ring topology, nodes serve as repeaters, receiving and re-transmitting incoming information.

Various approaches have been developed to avoid conflicts between nodes using a shared medium in a network. For example, in one common technique, a token-passing distributed-control scheme is used, where permission to transmit is passed sequentially from one node or station to another by means of a "token," a specified bit or flag set in an information frame, or a specifically-defined control frame. Alternatively, a node currently engaged in transmitting over the medium may pass control to the next node upon conclusion of its transmission by setting a token bit in its transmitted frame. A node recognizing the token bit is then free to transmit its own information if it so desires. Thus, multiple tokens may simultaneously exist on the medium. In another conventional approach, a currently-transmitting node passes control to the next node (i.e., issues a token) only on the return of its own frame. In this case, there is only one token available at any one time, simplifying management of the network. Both bus and ring topologies may be used in conjunction with the token-passing approach. In the case of a bus, nodes are numbered in succession to permit unique identification of which node may next receive the token. In the case of a ring, the "next" node is implicit in the direction of transmission.

In the simplest mode of operation, each node on the ring receives each frame packet and then passes it on (retransmits it) to its neighbor. If a particular node recognizes the packet destination address as being its own, that node copies the frame in addition to retransmitting it. The original transmitting source node takes its own frame off the ring as it returns from one cycle around the ring. In the single-token procedure, a sending node passes control to the next node by issuing a token after receipt of its own frame packet. A node with a frame to transmit must wait until it receives the token before transmitting.

The time to transfer data from a source node to a destination node is typically used as a measure of network performance. The transfer time is dependent upon a number of factors, a significant one of which is ring latency or delay. There are two major contributors to ring latency: the propagation delay required for a frame packet to cycle once around the ring; and the delay required to retransmit a frame packet at each node on the ring. In general, reduced ring latency results in better network performance.

The effect of ring latency is compounded by the increased bandwidth capabilities provided in modern high-speed fiber optic systems. As high speed networks become faster due to the packing of bits closer together in the fiber, the time it takes for a single bit to traverse the fiber stays essentially the same. Thus, for example, it may take approximately the same time to exchange a message between applications on a high speed fiber optic network, which may be capable of operating at 2 Gb/s, as it does over a 10 Mb/s Ethernet network. However, the increased capability of the fiber optic network to send more bits per unit time, as well as the increased capability of nodes to perform more instructions per unit time, results in an increase in the relative delay in the network. That is, the number of instruction cycles that a source node must wait for a reply to its message increases as the node's CPU cycle time decreases. As a result, ring latency is becoming the largest contributor to the reduction of performance in distributed network systems.

This problem is exacerbated in widely distributed network systems since propagation delay increases with distance. As the nodes in a distributed network become more geographically distributed, the number of instruction cycles that a source node must wait for its packet to return, or for an answer to its message, increases. In addition, as the node-to-node distance increases in a geographically distributed network system, the propagation delay, and thus ring latency, becomes unpredictable. The unpredictability of distributed network systems is particularly problematic when the network is required to carry synchronous data such as pulse coded modulation (PCM) data commonly used in telecommunications networks. The unpredictable arrival of frame packets prevents the receiving node from accurately identifying the divisions between PCM samples, thereby inhibiting the transfer of synchronous data through the asynchronous network.

Furthermore, to ensure proper reception of information over a distributed network, local timing signals (i.e., clock signals) controlling a given destination node must be precisely matched to those of the source node. However, despite being designed to operate at the same frequency, timing variations inevitably exist among network components. High frequency variations, referred to as jitter, are typically reduced to manageable levels through the use of jitter filters in each node. Low frequency variations, referred to as wander, are typically dealt with through the use of buffers located within the nodes of the network. Specifically, these buffers store a small amount of data, allowing it to build up

or be drained by small-magnitude wander without data loss or errors. When wander exceeds the capacity of the buffers, they either repeat (i.e., underflow) or discard (i.e., overflow) blocks of data to compensate for differences in timing between the source and destination nodes. Underflow and overflow conditions, generally referred to as slip, typically result in errors within the network. For example, in a voice circuit, slip may appear as popping or clicking sounds, whereas in data transmissions, slip is manifested by the loss of data. Very large buffers can reduce the probability of such errors, but they increase the delay through the network. Delay is undesirable, so buffer size is generally minimized.

Various techniques have been developed to maintain network synchronization and avoid such loss of data. For example, conventional synchronization techniques often require transmission of timing information through the network along with the data. A clock recovery system residing in a destination node uses the transmitted timing information to recover the frequency of the source node's clock and to generate a transmit clock having a frequency at which the destination node transmits the data to a destination user process. In addition, the recovered clock and data are provided to other nodes in the network. Regardless of the recovery technique, each node employs a phase-locked loop or other feedback circuitry that varies around the source node's clock frequency, continually adjusting to maintain lock on that frequency. This continual adjustment around the desired frequency causes jitter. As each subsequent node attempts to recover the clock, the jitter from all previous recovery attempts is accumulated. Eventually, this accumulated jitter may become too large, thereby resulting in data loss.

Another drawback to conventional clock recovery systems is that they are based upon the assumption that identical network reference clocks are provided to the source and destination nodes. This is often not the case in geographically-distributed telecommunications systems. It is not uncommon for each portion of a geographical-distributed telecommunications network to be synchronized to a different reference clock. Although those local clocks may be referenced to stratum 1 clocks, they may exhibit a phase difference over time that continues to increase until a slip in inter-nodal communications occurs. Moreover, if a network element such as a digital cross connect fails, certain network nodes may lose their reference clock. These nodes must then utilize their internal clocks, resulting in an increased loss of data due to the difference in phase and frequency between such nodes' internal clocks and the reference clocks.

What is needed, therefore, is a means for ensuring that the ring latency in a distributed network system is reliably controlled so as to support the transmission of synchronous data. In addition, the system must be capable of compensating for differences between source and destination nodes' clocks without loss of data and without causing excessive delays in the transmission of information across the network.

SUMMARY OF THE INVENTION

In brief summary, the present invention provides a distributed synchronization system for use in connection with a distributed, asynchronous, telecommunications network system that continually monitors and controls the flow of data through an implementing network node so as to prevent dataflow errors due to phase and frequency differences in source and destination nodal clocks.

Specifically, the present invention includes a synchronization data first-in-first-out (FIFO) memory for storing predetermined fields or portions of fields of a unique frame packet. The frame packet includes a frame synchronization field which marks the beginning of the frame packet; a payload field containing valid data; and a dead zone field which the present invention utilizes to perform synchronization functions. A frame synchronization subsystem of the present invention, implemented in a designated master node, operates such that a frame is released at the beginning of an independently-determined frame cycle regardless of the ring latency of the network.

A word resynchronization subsystem manages the flow of data through the data FIFO of each non-master node, receiving and storing the data at the source node's clock rate and re-transmitting the data at its own clock rate. Thus, the word resynchronization subsystem controls the operation of the synchronization data FIFO to effectively absorb any phase difference between the clocks of a source node and a destination node implementing the invention. A write controller, operating at the source node clock rate, and a read controller, operating at the destination node clock rate, asynchronously manage the passage of data through the data FIFO to maintain the level of data in the FIFO within an optimal range. During the receipt of a predetermined portion of a frame transmission which contains no valid data, the FIFO read and write controllers may temporarily suspend read and/or write operations from/to the FIFO to maintain the data level within the optimal range for efficient data transfer across the network.

Advantageously, the word resynchronization subsystem of the present invention prevents data FIFO overflow and underflow conditions from arising, thereby ensuring substantially error-free transmission through the implementing network node. A significant feature of the word resynchronization subsystem is its anticipatory nature, enabling it to compensate for phase differences between clock signals before data is lost. Another feature provided by the invention is the ability to momentarily control either or both the reading and writing of data from and to the FIFO to recover from clock deviations without loss of data and without causing substantial delays in the network.

The frame synchronization subsystem, on the other hand, buffers only the payload and frame synchronization fields of the frame packet, which are held until the beginning of a frame cycle, as determined by the occurrence of a local frame synchronization pulse. In response to that pulse, the frame synchronization subsystem transmits the frame synchronization and payload fields, onto which this subsystem appends a new, locally-determined dead zone to create a frame packet that is precisely one frame in length. This aspect of the present invention advantageously ensures that every non-master node receives a frame packet at predetermined intervals of time. In turn, such predictable and periodic receipt of frame packets enables the receiving nodes to precisely determine the boundaries between synchronous data samples contained within the frame packets. Furthermore, the frame synchronization subsystem automatically adjusts the amount of data buffered in the master node, dynamically self-adjusting to changes in the ring latency which may be caused, for example, by the addition (or deletion) of nodes to the network.

BRIEF DESCRIPTION OF THE DRAWINGS

The above and further advantages of the invention may be better understood by referring to the following description in conjunction with the accompanying drawings in which:

FIG. 1 is a block diagram of an expandable telecommunications system which employs a ring-type inter-nodal network to transfer information between nodes, all of which is constructed in accordance with a preferred embodiment of the present invention;

FIG. 2 is a block diagram of one type of programmable switching node that may be used in the telecommunications system of FIG. 1;

FIG. 3 is a block diagram of the nodal switch incorporated in the switching node illustrated in FIG. 2;

FIG. 4 is a block diagram of the distributed synchronization system of the present invention;

FIG. 5 is a block diagram showing the novel frame structure utilized by the distributed synchronization system of the present invention;

FIG. 6 is a block diagram showing the word resynchronization subsystem of the distributed synchronization system of the present invention;

FIG. 7 is a state diagram illustrating the functions performed by the write controller of the word resynchronization subsystem of the present invention;

FIG. 8 is a state diagram illustrating the functions performed by the read controller of the word resynchronization subsystem of the present invention;

FIG. 9 is a block diagram showing the frame synchronization subsystem of the distributed synchronization system of the present invention;

FIG. 10 is a state diagram illustrating the functions performed by the write controller of the frame synchronization subsystem of the present invention; and

FIG. 11 is a state diagram illustrating the functions performed by the read controller of the frame synchronization subsystem of the present invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

A. System Environment

FIG. 1 shows a large capacity, expandable, fully programmable telecommunications switching network system **100**. The network system **100** includes a series of programmable nodes **102** interconnected by a ring-architecture, inter-nodal network **110**. The programmable nodes include a master node **102***a*, programmable switching nodes **102***b*, **102***d*, and a voice processing resources node **102***c*. A host link **114** connects node **102***b* in communicating relationship with a host computer **104**. Nodes **102***a*, **102***c* and **102***d* may be controlled by host computer **104**, whether by additional host links to such nodes, by passing control information over inter-nodal network **110** or by separate host devices. Although only a single host computer **104** and host link **114** are shown for purposes of improved clarity, use of a local area network (LAN) to provide host/node communications permits multiple hosts to control the system **100** (or parts thereof) by configuring each host as a "client" and each node as a "server."

The nodes may perform any number of functions. For example, nodes **102***b* and **102***d* are switching nodes and include desired network/line interfaces for connection, respectively, with a public switched telephone network (PSTN) or a private network **106** and **118**, respectively. The term "private network" is intended in a broad sense to refer to any network or line or other interface other than the PSTN. Network/line interfaces **108**, **116** may terminate either digital networks or analog trunks/lines, or combinations of both types.

Node **102***a* is nominally designated a "master node," the significance of which is described below. As noted below, any of nodes **102***a*–**102***d* may be configured as the active master node. However, at any given time, there may be only one active master node.

Inter-nodal network **110** provides for high speed, high bandwidth digital communications among nodes **102***a*–**102***d*. Inter-nodal network **110** may be implemented using one or more fiber optic rings which enable each of the nodes to exchange packetized information with each other node served by network **110**. Inter-nodal network **110** may also be implemented with any of a variety of other types of communications networks, including Ethernet or other types of LANs, wireless communications networks or the PSTN (ATM/SONET). Using the PSTN for inter-nodal network **110**, for example, permits the nodes to be geographically distributed over large areas. Furthermore, other inter-nodal network topologies, such as a bus topology, are contemplated by the present invention.

The overall operation of system **100** is controlled by host **104**, which is commonly implemented with a personal computer (PC), workstation, or other computer on which a user's application software runs. Host **104** and node **102***b* exchange messages over host link **114**. Such messages are typically used to configure the node as well as direct call processing functions such as making connections and providing communications services (i.e., tone detection, tone generation and conferencing). Descriptions of exemplary programmable network protocols and communications services supported by nodes **102**, as well as the development of such protocols, may be found in commonly owned U.S. Pat. No. 5,426,694 to Mark P. Hebert, entitled "Telecommunications Switch Having Programmable Network Protocols and Communications Services," and U.S. patent application of Mark P. Hebert, entitled "Telecommunications Switch Having A Universal Applications Program Interface," filed on Nov. 30, 1995, Ser. No. 08/566,414.

FIG. 2 shows the major functional components that may be contained in one type of node which may be used in system **100**, programmable switching node **102***b*. Digital or analog network/line interfaces **206** are terminated on a series of line card input/output (10) cards **204**. A series of digital network T1, E1, J1 or analog trunk/line line cards **208** communicate with line card IO cards **204** over line card (LC) IO lines **210**. Line cards **208** are also interfaced with redundant switching buses **212***a* and **212***b* (collectively and generally referred to as switching buses **212**). Other types of network/line interfaces (and related line cards) such as DS3, SONET, SS7, ISDN or others may also be provided.

Diverse communications services such tone detection and generation, conferencing, voice recorded announcements, call progress analysis, speech recognition, ADPCM compression and many others are provided by one or more multifunction digital signal processing (MFDSP) cards **214**. Details of the architecture and operation of MFDSP cards **214** and other optional cards, as well as buses **212**, are disclosed in commonly owned U.S. Pat. No. 5,349,579. A ring (network) IO card **216** serves as an interface between inter-nodal network **110** and a nodal switch **202** of the present invention. A host interface may be provided as noted above to establish a communication link with host **104**. It should be understood that other cards may be added to or removed from the illustrative switch **102***b*.

In contrast to switching node **102***b*, voice processing resource node **102***c* (FIG. **1**) does not necessarily include line cards **208** and line card IO cards **204** since such a node need not interface with a PSTN or other network. However, such nodes may include additional components, such as a standard voice processing buses for communicating with,

for example, voice processing resources. For example, Dialogic Corporation of New Jersey produces a family of voice processing resource boards or cards which plug directly into certain standard voice processing buses and may be used in diverse applications including voice mail, fax mail, interactive voice response and others.

The detailed construction of a preferred embodiment of nodal switch **202** and ring IO card **216** of the present invention is shown in FIG. **3**. A central processing unit (CPU), with associated RAM/ROM, **302** is connected in communicating relationship with an address/data bus **304**. CPU **302** is also connected in communicating relationship with an HDLC bus (part of switching buses **212**) for communication with other cards within the node and may, depending upon the configuration of nodal switch **202**, also be connected in communicating relationship with host **104**. A data transmitter **306** and data receiver **308** are connected in communicating relationship with address/data buses **304** and a packet handling circuit **312**.

A high speed data receiver **314** is physically interfaced with inter-nodal network **110** for receiving information in the form of optical signals from that ring. Receiver **314** is preferably implemented with a Hewlett-Packard Company HDMP-1014 receiver chip, which is an emitter coupled logic (ECL) device. Conversion circuit **316** is connected to receive the output signals of receiver **314** and produce output signals that are compatible with transistor-transistor logic (TTL). The output of conversion circuit **316** is applied, through a buffer **318**, to the distributed synchronization system **326** of the present invention. The output of system **326** is applied to packet handling circuit **312** that transfers data to/from data receiver **308** and data transmitter **306**, respectively. A buffer **328**, conversion circuit **330**, and high speed data transmitter **332** perform functions which are complementary to those of buffer **318**, conversion circuit **316** and data receiver **314**, respectively. Transmitter **332** is preferably implemented with a Hewlett-Packard Company HDMP-1012 transmitter chip.

Receiver **314** includes circuitry which recovers a source node's clock signal from a received transmission and distributes it as a recovered ring clock **322** to the components of the nodal switch **202** dedicated to receiving frame packets, including the distributed synchronization system **326** of the present invention. A clock processor **334** that generates a local frame synchronization pulse **324** for use by the distributed synchronization system **326**. Local frame synchronization pulse **324** is derived from a network reference clock provided to the implementing node, typically from the PSTN or a private network.

A ring oscillator **336** generates a local ring clock **320** used by the components of nodal switch **202**, including distributed synchronization system **326**, to transmit frame packets. Further details of the structure and operation of nodal switch **202** may be found in commonly owned U.S. Pat. No. 5,544,163.

B. Distributed Synchronization System

1. In General

To ensure that the ring latency of the distributed network system **100** does not interfere with the transmission of synchronous (PCM) data, the present invention operates to delay the retransmission of a given frame packet until the occurrence of a predetermined frame synchronization signal (pulse), thereby dynamically adjusting the ring latency to an integer number of frame cycles. To avoid dataflow errors due to clock deviations between asynchronously-communicating nodes, the present invention performs the receive and retransmit functions asynchronously, maintaining the data throughput at an optimal rate to prevent dataflow errors while avoiding excessive increases in ring latency.

For the purpose of maintaining both word and frame synchronization between geographically-distributed nodes of an asynchronous network, the present invention comprises two related, yet functionally distinct, subsystems: a frame synchronization subsystem operating in a designated master node; and a word resynchronization subsystem operating in the remaining (non-master) nodes. Each subsystem works in conjunction with a unique frame packet discussed in detail below.

Referring now to FIG. **4**, the preferred embodiment of the distributed synchronization system **326** comprises a synchronization data FIFO **402**, a word resynchronization subsystem **404** and a frame synchronization subsystem **406**. Asynchronous data in the form of frame packets **502** (FIG. **5A**) arrive on data input line **401** and are written into data FIFO **402**. Data which is read from data FIFO **402** appears on a data output line **403**.

Word resynchronization subsystem **404** monitors, via line **412**, frame packets presented to the input of data FIFO **402**. Subsystem **404** also monitors the level of data in the data FIFO **402** and controls the writing of the presented frame packets into the data FIFO **402** via a control/status line **408**. Subsystem **404** also receives as inputs recovered ring clock **322** to control the writing of data into the data FIFO **402**; and a local ring clock **320** to control the reading of data from data FIFO **402**.

Frame synchronization subsystem **406** also monitors the frame packets presented to the data FIFO **402**. Subsystem **406** also monitors the level of data in the data FIFO **402** and controls the reading and writing of frame packets into the data FIFO **402** via the control/status line **410**. Since subsystem **406** also asynchronously performs the reception and retransmission of frame packets, it receives as inputs the recovered ring clock **322** and local ring clock **320**, both of which are utilized for similar purposes as in the word resynchronization subsystem **404**. In addition, subsystem **406** receives the local frame synchronization pulse **324** that it uses to transmit the frame packets from FIFO **402**.

FIG. **5A** shows a general structure for a frame packet **502** for exchanging information over the inter-nodal network **110**. Each frame packet **502** comprises a number of fields, each containing one or more words of data, control information or fill frames (i.e., non-data). A frame synchronization field **504** provides an indication of the beginning of a frame packet **502**. A payload field **506** comprises a number of subpackets, each containing data for transmission among nodes served by inter-nodal network **110**. Payload field **506** may contain any type of data within its subpackets, including circuit switched data, packet switched data, voice processing data and others. A dead zone field **508**, which does not contain valid data and whose length or duration is dynamically-adjustable, is used for synchronization purposes as described below. Additional information regarding the structure of the subpackets as well as details of various packet structures for transferring different types of information, is described in commonly owned U.S. Pat. No. 5,544,163.

FIG. **5B**, in conjunction with FIG. **5A**, illustrates a preferred approach for allocating the bandwidth of inter-nodal network **110** for the purpose of transferring data among nodes. Transfer of data over the network is preferably made within framing windows **510**, each of which is 125 μs in duration. A period of 125 μs is preferred since it corresponds with the sampling rate (8 kHz) of most widely used circuit switched network protocols, meaning that the values

of circuit switched data may change every 125 μs. Thus, by requiring that all inter-nodal transfers of circuit switched data take place in less than 125 μs, inter-nodal network **110** ensures that all such data is transferred before any value changes. This also permits inter-nodal network **110** to operate asynchronously with respect to the PSTN or private networks **106,118** (FIG. 1).

2. Word Resynchronization Subsystem

With reference now to FIGS. **6–8**, the detailed operation of word resynchronization subsystem **404** is described. FIG. **6** is a functional block diagram of the word resynchronization subsystem **404** and data FIFO **402** of the distributed synchronization system **326** of the present invention. The word resynchronization subsystem **404** generally includes a write controller **602**, a read controller **604**, an initializer **606** and a counter **608** connected in communicating relationships by a control bus **626**. Write controller **602** controls the writing of data into the data FIFO **402** via a write enable (WEN) signal line **610**. Read controller **604** controls the reading of data from data FIFO **402** via a read enable (REN) signal line **612**.

Write controller **602** and read controller **604** control the flow of data through data FIFO **402** in response to a time-varying data level **620** of the FIFO as well as which field of a frame packet **624** is currently present at the input of data FIFO **402**. More specifically, the controllers **602,604** cooperate to maintain data level **620** between an upper threshold level **618** and a lower threshold level **622**. Thresholds **618,622** are preferably chosen to define an optimal range of data level **620** to absorb phase differences between the clocks of the source node and the destination node without contributing excessively to ring latency. Upper threshold level **618** is preferably represented by a programmable almost full (PAF) flag **614**, while the lower threshold level **622** is preferably represented by a programmable almost empty (PAE) flag **616**. Together, the two flags **614,616** provide a current indication of the data level **620**.

PAE flag **616** indicates when data level **620** is below the associated lower threshold level **622**. Similarly, PAF flag **614** indicates when data level **620** is above the associated upper threshold level **618**. When data level **620** is at or below the lower threshold **622**, PAE flag **616** is in its active state whereas when the data level **620** is at or above the upper threshold **618**, PAF flag **614** is in its active state. Alternatively, when data level **620** is above lower threshold **622** and below upper threshold **618**, the PAE and PAF flags are both inactive. Thresholds **618** and **622** are initially set by initializer **606** via initialization control line **628** to predetermined levels.

As noted above, frame packet **624** comprises a number of fields. The operations performed by word resynchronization subsystem **404** depend, in part, upon which field of the frame packet **624** is present at the input to data FIFO **402**. This is determined by counter **608** via monitor line **412**. Upon receipt of a frame synchronization field counter **608** begins to count the number of words received by data FIFO **402** and resets an internal timer. When counter **608** reaches a predetermined value corresponding to the beginning of the dead zone field **508**, the counter transmits a signal on control bus **626** instructing write controller **602** and read controller **604** to resynchronize.

Referring now to FIGS. **6** and **7**, write controller **602** preferably operates in accordance with a state machine **700** having three states: a reset state **702**, a write suspend state **704** and a write state **706**. Initially, write controller **602** is at reset state **702**. Write controller **602** may be reset for any number of reasons, such as upon receipt of power, when data FIFO **402** is cleared, when a node is initialized prior to commencing communications over the network and the like.

At reset state **702**, write controller **602** initially sets the WEN control line **610** inactive, thus preventing the writing of data into data FIFO **402** until the necessary conditions are present. Upon completion of these initialization procedures, write controller **602** advances to write suspend state **704** as shown by state transition line **708**.

While write controller **602** is at write suspend state **704**, it maintains WEN control line **610** inactive while monitoring PAF flag **614** and control bus **626**. If PAF flag **614** is active, then data level **620** is above the upper threshold **618** as discussed above. Under such conditions, write controller **602** continues to remain in the write suspend state **704** unless and until either a frame packet is received or data level **620** falls below upper threshold **618**. When data level **620** is below upper threshold **618** and PAF flag **614** is thus inactive, write controller **602** will allow data to be written into the data FIFO **402**. Thus, when a frame packet is presented at the input of data FIFO **402** or when data FIFO **402** is not almost full, then write controller **602** advances to write state **706** as shown by state transition line **710**.

At write state **706**, write controller **602** sets WEN line **610** active to enable writing of data into data FIFO **402**. The writing of data continues until two conditions simultaneously occur. If counter **608** (via control bus **626**) indicates that dead zone field **508** is currently present at the input to data FIFO **402** which means that the payload field **506** has been completely written into the FIFO) and the data level **620** is above the upper threshold **618** (and is thus above the desired optimal range), the writing of data is suspended. Thus, if payload field **506** of the current frame packet is completely written into FIFO **402** and the FIFO is almost full, write controller **602** advances to write suspend state **704** as shown by state transition line **712**.

Referring now to FIGS. **6** and **8**, read controller **604** preferably operates in accordance with a state machine **800** having three states: a reset state **802**, a read suspend state **804** and a read state **806**. Initially, read controller **604** is at reset state **802**. Read controller **604** may be reset for any number of reasons such as those discussed above with respect to write controller reset state **702**. At reset state **802**, read controller **604** sets REN control line **612** inactive to prevent the reading of data from FIFO **402** until the necessary conditions are met. Upon completion of the reset/initialization procedures, read controller **604** advances to read suspend state **804** as shown by state transition line **808**.

While read controller **604** is at read suspend state **804**, it maintains REN control line **812** inactive while monitoring PAE flag **616** and control bus **626**. If PAE flag **616** is active, then data level **620** is below lower threshold **622** as discussed above. Under such conditions, the read controller **604** continues to remain in read suspend state **804** unless a frame packet is received or data level **620** rises above lower threshold **622**. When either of those events occurs, read controller **604** advances to read state **806**.

At read state **806**, read controller **604** sets REN control signal **612** active to enable reading of data from FIFO **402**. So long as counter **608** indicates, again, via control bus **626**, that dead zone field **508** is currently present at the input to the data FIFO **402** (i.e., the payload field **506** has been completely written into the FIFO) or data level **620** is in the optimal range, reading of data remains enabled and data will continue to be read from data FIFO **402**.

However, if counter **608** indicates that dead zone field **508** is currently present at the input to data FIFO **402** and data level **620** is simultaneously below lower threshold **622** (and

is thus below the optimal range), the reading of data is suspended. Thus, if payload field **506** of the currently-presented frame packet is completely written into FIFO **402** and the FIFO **402** has become almost empty, read controller **602** advances to read suspend state **804** as shown by state transition line **812**.

Write and read controllers **602,604** interoperate to maintain data level **620** within between upper threshold **618** and lower threshold **622**. As noted above, write and read controllers **602,604** perform their functions in response to an internally-generated synchronization instruction based upon the states of PAF and PAE flags **614,616** and the portion of frame packet **502** present at the input of data FIFO **402**.

Upon receipt of a frame synchronization field counter **608** begins counting the data words which follow that field (i.e., the words in payload field **506**). When counter **608** reaches a predetermined number of words representing the end of payload field **506** (and the beginning of dead zone field **508**), counter **608** issues a resynchronization signal over control bus **626** which causes write and read controllers **602,604** to check the status of the PAE and PAF flags. If another frame packet is not subsequently received, counter **608** continues to reset and increment, each time generating a resynchronization instruction which causes word resynchronization subsystem **404** to resynchronize. In other words, counter **608** begins incrementing when a frame synchronization field **504** is received and, upon reaching its maximum predetermined value, counter **608** resets and begins again in anticipation of receiving another frame synchronization field **504**. Thus, resynchronization is guaranteed to occur regardless of whether a frame synchronization field **504** is actually received.

If, during resynchronization, write and read controllers **602,604** determine that data level **620** is between upper and lower thresholds **618,622**, then controllers **602,604** will allow continuous reading and writing of data through the data FIFO. However, if data level **620** is either above upper threshold **618** or below the lower threshold **622**, then read and write controllers **602,604** will momentarily suspend writing or reading of data as needed to restore data level **620** to within the optimal range.

Upon initialization or invocation of a reset condition, write and read controllers **402,404** are placed in reset states **702,802**, respectively. While the controllers are in their respective reset states, initializer **606** sets the values of upper threshold **618** and lower threshold **622** at respective predetermined values to define an optimal range for data level **620**. By setting those thresholds, a desired optimal range for data level **620** is established prior to the presentation of valid data at the input of data FIFO **402**.

It is important that data level **620** is built-up to within the established optimal range before valid data is received to avoid the immediate slippage of words (i.e., during the time delay that would be incurred in filling a completely empty FIFO to at least the almost empty level). During periods when no nodes are transmitting onto the network, the master node may generate fill frames or frame packets having payload fields **506** that contain no valid data. Such fill frames may be received and stored by all the nodes in the network implementing the word resynchronization subsystem **404** of the present invention. Thus, although no valid data is received, the fill frames are nonetheless stored in the data FIFO **402**, thereby maintaining data level **620** in the optimal range prior to the receipt of valid data (i.e., prior to normal communications).

Preferably, an optimal range for data level **620** is chosen based upon the expected deviations between the source and destination node clocks in the network. In a preferred embodiment of the present invention, each node has a local oscillator which operates at 131.072 MHz±25 ppm (parts per million). The local clock of each node is preferably configured to operate at half the oscillator rate. Thus, the tolerance of each node's local clock is (131.072+2×25) or 1638 words/second. That is, variations in the frequency of each node's local clock cause corresponding variations in data transmission by as much as 1638 words/second. The largest difference between two nodes on the network will occur when a node containing a local oscillator running at 131.072+25 ppm is communicating with another node containing a local oscillator running at 131.072−25 ppm. This worst case scenario will result in an error condition of (2×1638) or 3276 words/second. In that scenario, a slip of one word can be expected every 305.25 μs.

Note, however, that in addition to instructing the controllers to resynchronize upon receipt of the dead zone, counter **608** also instructs the write and read controllers to resynchronize when no frame synchronization field **504** is received at all. That is, counter **608** continually resets and increments (up to the expiration of its internal 125 μs timer) regardless of whether a frame synchronization field **504** appears at the input of the data FIFO **402**. This periodic resynchronization continues indefinitely and is interrupted only when a frame synchronization field **504** is received. Thus, the longest period of time between resynchronizations is when a frame synchronization field **504** is received immediately prior to the timer expiring. For example, if the packet frame contains 8125 words and lasts 125 μs and dead zone field **508** is 100 words in length, then the timer will indicate the beginning of the dead zone at (8125−100 words)×15 nanoseconds) or 120.4 μs, with the longest duration between resynchronizations being (120.4×2) or 240.8 μs.

Because word resynchronization is preferably performed every frame, or once every 125 μs, which is less than the time of an expected word slip of 305 μs, no word slips should occur. As a result, an optimal data FIFO level **620** may be only a few words. In a preferred embodiment, upper threshold **618** and lower threshold **622** are determined based upon how often data FIFO **402** will be serviced by the controllers, the anticipated deviations between the source and destination clocks, the hysteric behavior of the subsystem, as well as the data FIFO's sensitivity to underflow and overflow conditions. However, as one skilled in the relevant art would find apparent, other criteria may also be considered. In a preferred embodiment, the capacity of data FIFO **402** is significantly larger than upper threshold **618**, to provide protection against overflow conditions. However, if the data FIFO level becomes too large, there will be significant delay through the node. Although it is desirable to maintain the data level **620** as low as possible to reduce the delay through the node, to provide protection against underflow conditions there must be a data level sufficient to prevent loss of data in light of the above factors. These concerns are balanced against ring latency requirements to obtain an optimal range for data level **620**. In a preferred embodiment, the optimal data level **620** is set at an 8 word depth, with the associated upper and lower thresholds **618, 620** set at 8 and 12 words, respectively. A data FIFO level of 8 words will not cause significant delay through the node (8×15 ns=120 ns), while providing a conservative number of words to prevent slippage (despite the fact that none should occur given the above calculations).

Note that writing of data is suspended only during the receipt of the dead zone field **508** when only fill frames

(non-valid data) are received. As a result, some, all, or none of the dead zone field **508** will be stored in the data FIFO **402**. The portion that is stored is the amount necessary to guarantee that the frame packet which is subsequently read from the data FIFO **402** and transmitted at the rate of the local clock will average 125 μs in length. Furthermore, the clock signal transmitted to another node will not contain jitter or wander components of the recovered source node's clock.

3. Frame Synchronization Subsystem

With reference now to FIGS. **9–11**, the operation of frame synchronization subsystem **406** is described. FIG. **9** is a functional block diagram of subsystem **406** and data FIFO **402** of the distributed synchronization system **326** of the present invention. Frame synchronization subsystem **406** includes a write controller **902**, a read controller **904**, an initializer **906**, and a counter **908**. Write controller **902** controls writing of data into data FIFO **402** via a write enable (WEN) signal line **910**. Read controller **904** controls reading of data from data FIFO **402** via a read enable (REN) signal line **912**. When the REN and WEN control lines are active, the data FIFO **402** is enabled to read and write data, respectively.

A lower data threshold **922** is represented by a synchronization programmable almost empty (SYNCPAE) flag **916**. When data level **920** is at or below lower threshold **922**, SYNCPAE flag **916** is in its active state. Conversely, when data level **920** is above lower threshold **922**, the SYNCPAE flag is in its inactive state. Lower threshold **922** is initially set by an initializer **906** via initialization control line **928** to a predetermined level (discussed below).

Frame synchronization subsystem **406** generates an initialization frame when the network is first initialized. Each non-master node receives and retransmits the initialization frame, causing each node to successively initialize its nodal switch. When the initialization frame returns to the master node, the master node itself then initializes with the knowledge that all other nodes in the network are ready to commence inter-nodal communications. The master node then transmits a frame synchronization field designating the frame boundaries around the ring.

The determination of which field of frame packet **901** is present at the FIFO input is made by a counter **908** via a monitor line **412**. When counter **908** detects a frame synchronization field **903**, it counts up to the dead zone field **907** then issues a self-synchronization command to write controller **902** to begin or suspend write operations. When counter **908** does not detect a frame synchronization field **903** (and thus a dead zone field **907** as well) during a frame transmission, the counter issues a self-synchronization command to write controller **902**. Based upon the frame packet field and the invocation of self-synchronization, counter **908** generates a signal on control bus **926** instructing write controller **902** to write or not write the received fields into data FIFO **402**.

Frame synchronization subsystem **406** dynamically adjusts the amount of data stored in data FIFO **402** to accommodate variations in ring latency. If, for example, the ring latency is increased suddenly due to a node failure and subsequent loop-back operations, then data level **920** in data FIFO **402** will either rise or fall depending upon the time of the failure in relation to the frame that is being processed by the FIFO. However, frame synchronization subsystem **406** automatically recovers because the next frame synchronization field **903** that appears at the output of data FIFO **402** does so significantly before the occurrence of the local frame synchronization pulse **324**. As a result, reads are suspended while data FIFO **402** is filled with data, thereby automatically raising the data level by the amount that it was previously depleted.

Referring to FIGS. **9** and **10**, write controller state machine **1000** has three states: a reset state **1002**, a write suspend state **1004**, and a write state **1006**. Initially, write controller **902** is at reset state **1002**.

Write controller **902** may be reset for any number of reasons, such as when initializer **906** sets the lower threshold level **922**, upon the initial application or an interruption of power to the frame synchronization subsystem **406**, and the like.

At reset state **1002**, write controller **902** sets the WEN control line **910** inactive to prevent the writing of data into data FIFO **402** until the necessary conditions are met. Upon completion of the initialization procedures, write controller **902** advances to write suspend state **1004** as shown by state transition line **1012**.

While write controller **902** is at write suspend state **1004**, it maintains WEN control line **910** inactive to prevent writes from occurring. During this time, counter **908** monitors the input of the data FIFO **402** to determine which portion of frame packet **901** is currently present. When counter **908** detects a frame synchronization field **903** or does not detect a frame packet during a frame transmission, counter **908** invokes a self-synchronization operation, instructing write controller **902** to begin writing data into the data FIFO **402**. When this occurs, write controller **902** transitions from write suspend state **1004** to write state **1006** as shown by state transition line **1014**.

At write state **1006**, write controller **902** sets WEN control line **910** active to begin writing the contents of any received frame packets into data FIFO **402**. The writing of the frame packet fields continues until counter **908** indicates that dead zone field **907** is currently present at the input to the data FIFO **402**. On the other hand, if counter **908** did not detect a frame packet during a frame transmission, then write controller **902** will write fill frames into data FIFO **402** while in write state **1006**. In this circumstance, the counter **908** will still indicate the point at which a dead zone **907** would normally have appeared at the input of data FIFO **402**.

In other words, regardless of whether a frame packet or fill frames are being written into data FIFO **402**, write controller **902** will transition to write suspend state **1004** during a portion of the frame transmission. Thus, when dead zone field **907** of the current frame packet **901** is detected at the input to data FIFO **402** or when self-synchronization occurs, write controller **902** advances to write suspend state **1004** as shown by state transition line **1010**. As a result, write controller **902** writes only the frame synchronization and payload fields (and not the dead zone field) into data FIFO **402**. If no frame packet is presented to data FIFO **402**, then write controller **902** periodically transitions between write state **1004** and write suspend state **1006**.

Referring to FIGS. **9** and **11**, read controller state machine **1100** has three states: a reset state **1102**, a read state **1104**; and a read suspend state **1106**. Initially, read controller **904** is at reset state **1102**. At the reset state **1102**, read controller **904** sets the read enable REN control line **912** inactive to prevent the reading of data from data FIFO **402** until the necessary conditions are present. Upon completion of the initialization procedures, read controller **904** advances to read state **1104** as shown by state transition line **1112**.

While read controller **904** is at read state **1104**, it sets the REN control line **912** active while it monitors the output of data FIFO **402**. This will maintain data FIFO **402** at an optimal minimal level, somewhere close to empty, until a

frame packet has been received and stored in the FIFO. When the controller **904** determines that frame synchronization field **903** is about to be read from the data FIFO **402**, read controller **904** advances to read suspend state **1106** as shown by state transition line **1114**.

In read suspend state **1106**, read controller **904** sets the REN control line **912** inactive to stop the reads from the data FIFO **402** from continuing. Read controller **904** then waits until the simultaneous occurrence of two conditions: the receipt of the local frame synchronization pulse **324** and the accumulation of data in FIFO **402** such that data level **920** is above the lower threshold **922**. If the local frame synchronization pulse **324** occurs and the data FIFO **402** is not almost empty, then there is a sufficient amount of data in the data FIFO **402** to safely read data without causing underflow conditions. When data level **920** is below lower threshold **922**, read controller **904** remains in the read suspend state **1106**. On the other hand, when the data level **920** is above the lower threshold **922**, the read controller **904** will allow data to be read from data FIFO **402** upon receipt of a local frame synchronization pulse **324**. Thus, when a frame packet **911** is presented at the output of data FIFO **402** and data level **920** is not almost empty, then read controller **904** advances to read state **1104** as shown by state transition line **1110**.

If a local frame synchronization pulse **324** occurs prior to the time the data level **920** exceeds lower threshold **922**, read controller **904** will remain in read suspend state **1106** and continue to accumulate data until the next occurrence of the local frame synchronization pulse **324**. Thus, when a the local frame synchronization pulse **324** occurs and data level **920** is above lower threshold **922**, read controller **904** transitions to read state **1104**, releasing frame synchronization field **903** and payload field **905**. When the next frame synchronization field appears at the output of data FIFO **402**, read controller **904** will return to the read suspend state **1106** until the next local frame synchronization pulse **324** occurs.

Write and read controllers **902,904** perform their functions asychronously with respect to each other. However, their functions are coordinated to ensure that a frame packet is released from the implementing master node **102***a* at such time that the ring latency will be an integer multiple of frame packets. Write and read controllers **902**, **904** maintain an optimal amount of frame packet fields in data FIFO **402** so that a frame packet is ready for transmission upon the occurrence of a pulse while not causing excessive delay through the master node or exposing the node to potential underflow conditions. This coordinated operation of the write and read controllers is discussed below.

When write and read controllers **902,904** are in their respective reset states, WEN and REN control lines **910**, **912** are set inactive to prevent any data from being stored in, or retrieved from, data FIFO **402**. After initialization, write controller **902** advances to write suspend state **1004** and read controller **904** advances to read state **1104**. No data is written into the data FIFO **402** until either a frame synchronization field **903** is detected at the input of data FIFO **402** or a self-synchronization signal is received. However, data is read from data FIFO **402**, although initially no valid data will be presented at the output of the FIFO.

If a frame synchronization field **903** is not detected within a predetermined time equivalent to a frame transmission (i.e., 125 μs), then counter **908** generates a self-synchronization signal. This causes write controller **902** to advance to write state **1006**, while read controller **904** remains in read state **1104**. Thus, any data received will be written into and read from the data FIFO **402** with little or no delay since the data FIFO is essentially empty.

Since the write controller **902** and the read controller **904** operate asynchronously, data level **920** in the data FIFO **402** may drift upward. To avoid having an unnecessary number of fill frames in data FIFO **402** when a frame packet containing valid data is received, write controller **902** preferably periodically ceases writes operations while the read controller **904** continues read operations. That is, if a frame synchronization field **903** is not detected while the write controller **902** is in the write state **1006** and the counter **908** is incrementing, then write controller **902** transitions to the write suspend state **1004**, allowing read controller **904** to operate to reduce data level **920** during the time that dead zone field **907** is presented to data FIFO **402**. Write controller **902** will preferably transition between those two states indefinitely until a frame synchronization field **903** is received.

Upon receipt of a frame synchronization field **903**, counter **908** will reset and begin to count the number of words written into data FIFO **402**. Write controller **902** will either remain in write state **1006** or transition to write state **1006** from the write suspend state **1004**. When counter **908** reaches a predetermined value indicating that frame synchronization field **903** and payload field **905** have been written into data FIFO **402** and dead zone field **907** is currently present at the input of data FIFO **402**, then counter **908** instructs write controller **902** to cease writing into data FIFO **402** (since the received dead zone field **907** is not retransmited by the subsystem **406**).

When frame synchronization field **903** appears at the output of data FIFO **402**, read controller **904** advances to read suspend state **1106**, causing data FIFO **402** to start filling up. Upon the occurrence of a local frame synchronization pulse **324**, read controller **904** reads the frame synchronization field **903** and the payload field **905** from data FIFO **402**. Read controller **904** continues to read data from the FIFO until it detects a new frame synchronization field **903** presented at the output of the data FIFO **402**. When this occurs, the read controller ceases to read from the FIFO, causing fill frames to be added to the frame packet until it contains a sufficient number of words to precisely equal a frame transmission. Thus, a new dead zone **909** is effectively created by the ceasing of read operations. This guarantees that each frame packet **911** transmitted from the master node **102***a* contains an exact predetermined number of words for a frame transmission.

Since write controller **902** does not write (i.e., discards) the dead zone field **907** while read controller **904** is performing read operations, data FIFO **402** will be depleted by an amount of words approximately equivalent to the size of the dead zone field **907**. Likewise, when read controller **904** is in read suspend state **1106** while write controller **902** is in the write state **1006**, data level **920** will increase by the size of the dead zone field **907**.

Thus, to avoid dataflow errors from occurring, data FIFO **402** must be maintained with a number of words equal to at least the number of words contained in dead zone field **907**.

An additional requirement is that data FIFO **402** must contain a quantity of data, referred to as the "remainder", that is equivalent to the difference between the ring latency and an integer multiple of the frame. If the remainder is less than the size of dead zone field **907**, then the remainder plus a complete frame of data (payload field **905** and frame synchronization field **903**) is stored in the FIFO since the data level **920** will be below the lower threshold **922** when a pulse **324** occurs, causing the read controller **904** to stay in the read suspend state **1106** until a following pulse **324** after another frame of data has been written into the FIFO.

On the other hand, if the remainder is greater than the size of dead zone field **907**, then only the remainder will be stored in the data FIFO since the remainder number of words in the FIFO **402** will always be above the lower threshold **922** set at the size of the dead zone field **907**. As a result, SYNCPAE flag **916** will never become active once a frame packet has been received.

In a preferred embodiment of the present invention, the distributed synchronization system **326** comprises both a frame synchronization subsystem **406** implemented operating in a master node and a word resynchronization subsystem **404** implemented in each non-master node. However, as one skilled in the relevant art would find apparent, distributed synchronization system **326** may be implemented with either the frame synchronization subsystem **406** or the word resynchronization subsystem **404** individually.

It is noted, however, that a network implementing both subsystems enables continued, robust synchronized communications to occur between asynchronous nodes in the network while preventing dataflow errors and without contributing excessively to network latency.

As a result, in a preferred embodiment of the present invention, each network node is implemented with both subsystems. Accordingly, the size of FIFO **402** is preferably large enough to accommodate both of these implementations. Thus, FIFO **402** preferably has a byte capacity which is slightly larger than the amount of data contained within one frame. This will provide an implementing node with the capacity of acting as a master node and performing frame synchronization. Alternatively, the node may be a non-master node, performing solely word resynchronization. Furthermore, if such a node is implemented in a network system that is not distributed, frame synchronization is not required, thereby enabling the FIFO **402** to be considerably smaller.

While the invention has been particularly shown and described with reference to preferred embodiments thereof, it will be understood by those skilled in the art that various changes in form and details may be made therein without departing from the spirit and scope of the invention. Furthermore, the terms and expressions which have been employed are used as terms of description and not of limitation, and there is no intention, in the use of such terms and expressions, of excluding any equivalents of the features shown and described or portions thereof, but it is recognized that various modifications are possible within the scope of the invention claimed.

What is claimed is:

1. A distributed synchronization system for use in a node of an expandable telecommunications system including a plurality of nodes interconnected by an inter-nodal network, with one of said nodes being a master node, the system, comprising:

a synchronization data memory implemented in a master node and each of said nodes which are non-master nodes for storing predetermined portions of an inbound frame packet received from a source node and from which information is retransmitted to a destination node on the network;

a frame synchronization subsystem implemented in said master node, and connected in communicating relationship with said memory, configured to release an outbound frame packet at the beginning of an independently-determined frame cycle, based upon the occurrence of a local frame synchronization pulse, and wherein said frame packets include a frame synchronization field indicating the beginning of a new frame packet, a payload field containing valid data and a dynamically-adjustable dead zone comprising a non-valid data field; and

a word resynchronization subsystem implemented in said master node and each non-master node and connected in communicating relationship with said memory and configured to control storage of said predetermined portions using a recovered source node clock signal and to control retransmission of said information according to a local clock signal, such that each non-master node receives a frame packet at predetermined interval of time whereby dataflow errors due to phase differences in source and destination nodal clocks are substantially avoided.

2. The synchronization system of claim **1** wherein the frame synchronization subsystem comprises:

a counter for generating a signal indicative of which portion of the inbound frame packet is present at an input to said memory;

an initializer connected to said memory for establishing a threshold data level of said memory;

a read controller connected to said memory and responsive to a signal indicative of whether a current data level of said memory is greater than said threshold data level to alternately enable or suspend reading of information from said memory, a master clock signal for reading information from said memory for inclusion in said outbound frame packet, and a frame synchronization pulse which is derived from a network reference clock and is used to trigger the release of said outbound frame packet;

a write controller connected to said memory and responsive to said recovered source node clock signal to alternately enable or suspend writing of information into said memory;

said counter, initializer and read and write controllers connected in communicating relationships by a control bus.

3. The synchronization system of claim **1** wherein the word resynchronization subsystem further comprises:

a counter for counting a number of words that have been written into the data memory following receipt of said frame synchronization field and resetting a timer, for generating a signal indicative of which portion of the inbound frame packet is present at an input to said memory, and to instruct a read controller and a write controller to resynchronize when the counter reaches a predetermined value corresponding to the presence of said non-valid data field at said memory input;

an initializer connected to said memory for establishing upper and lower threshold data levels of said memory;

said read controller connected to said memory and responsive to a signal indicative of whether a current data level of said memory is greater than said lower threshold data level, and a master clock signal for controlling the reading of information from said memory;

said write controller connected to said memory and responsive to a signal indicative of whether a current data level of said memory is less than said upper threshold level, and a recovered source node clock signal for controlling the writing of information into said memory;

wherein said read and write controllers function in cooperating relationship to maintain the data level of said

memory within an optimal range between said upper and lower threshold levels;

said counter, initializer and read and write controllers are connected in communicating relationships by a control bus.

4. The synchronization system of claim 3 wherein said read and write controllers function to substantially compensate for phase differences between said recovered source node clock signal and said local clock signal by receiving and storing data in accordance with said recovered source node clock signal and retrieving and retransmitting the data in accordance with said local clock signal.

5. A telecommunications switch which is operable as a node in an expandable telecommunications system, said switch comprising:

one or more nodal switches for dynamically connecting or disconnecting communication paths with respect to various ones of a plurality of ports and transmitting and receiving packetized information over an inter-nodal network, said inter-nodal network for providing communications between said telecommunications switch and other nodes associated with said system; and

a word resynchronization subsystem including a memory and a controller, wherein said memory is implemented in a master node and each non-master node for storing predetermined portions of an inbound frame packet received from a source node and from which information for inclusion in an outbound frame packet is retrieved, and said controller for regulating a flow of information through said memory in response to a time-varying data level of the memory, and maintaining said data level within a predetermined optimal range; and

frame synchronization subsystem implemented in said master node, and connected in communicating relationship with said memory, configured to release an outbound frame packet at the beginning of an independently-determined frame cycle, based upon the occurrence of a local frame synchronization pulse, and wherein said frame packets include a frame synchronization field indicating the beginning of a new frame packet, a payload field containing valid data and a dynamically-adjustable dead zone comprising a non-valid data field.

6. A method of synchronizing data in a node of an expandable telecommunications system, the system comprising a plurality of switching nodes interconnected by an inter-nodal network with one of said nodes being a master node, said method comprising the steps of:

(a) at the master node, transmitting a first frame packet at the beginning of an independently-determined frame cycle:

(b) at each frame packet, a frame synchronization field indicating the beginning of said frame packet, a payload field including a number of subpackets, each containing data for transmission among nodes served by the inter-nodal network, and a dynamically-adjustable dead zone for synchronization purposes, comprising a non-valid data field;

(c) at a first non-master node, storing in a memory predetermined portions of said frame packet, and retrieving from said memory information which is to be retransmitted to another non-master node or the master node on the network, and controlling said retransmission of said information using a local frame synchronization pulse, such that each non-master node receives a frame packet at predetermined interval of time whereby data flow errors due to phase differences in source and destination nodal clocks are substantially avoided; and

(d) repeating step (c) at each non-master node until said first frame packet returns to said master node.

* * * * *

US006108346A

United States Patent [19]

Doucette et al.

[11] Patent Number: 6,108,346

[45] Date of Patent: Aug. 22, 2000

[54] **COMBINED SYNCHRONOUS AND ASYNCHRONOUS MESSAGE TRANSMISSION**

[75] Inventors: **John Doucette**, Londonderry; **Thomas J. Bryden**, Peterborough; **Todd Byron**, Manchester, all of N.H.

[73] Assignee: **Xiox Corporation**, Burlingame, Calif.

[21] Appl. No.: **09/268,099**

[22] Filed: **Mar. 13, 1999**

Related U.S. Application Data

[60] Provisional application No. 60/098,297, Aug. 27, 1998.

[51] **Int. Cl.**[7] **H04L 12/403**

[52] **U.S. Cl.** **370/450**; 370/353; 370/468

[58] **Field of Search** 370/352, 353, 370/354, 452, 460, 450, 468, 909, 470, 471, 472, 473, 476; 709/251, 238

[56] **References Cited**

U.S. PATENT DOCUMENTS

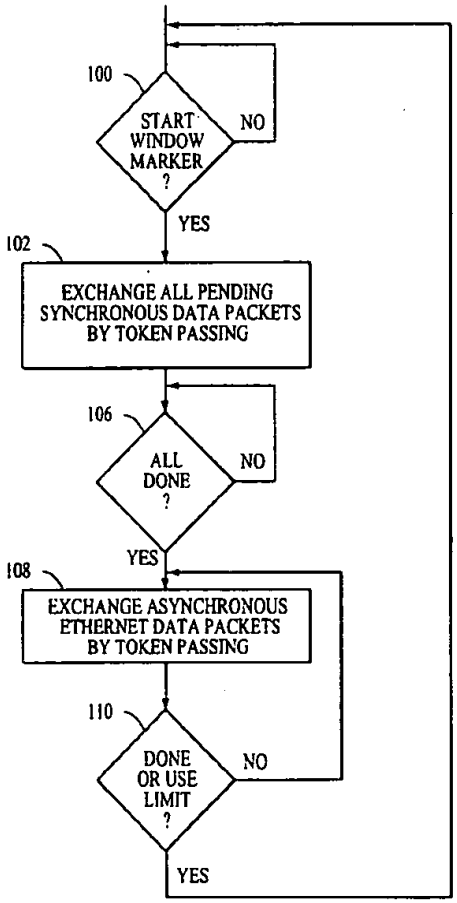
5,392,280	2/1995	Zheng	370/60
5,570,355	10/1996	Dial et al.	370/352
5,935,214	9/1999	Stiegler et al.	709/251

Primary Examiner—Huy D. Vu
Assistant Examiner—Kevin C. Harper
Attorney, Agent, or Firm—Elmer Galbi

[57] **ABSTRACT**

A communication system including a collection of modules coupled in a ring architecture which integrates synchronous and asynchronous message transmission. Asynchronous data packets and synchronous voice packets are exchange on a single communication link. Packetized information exchange references a fixed length window with all synchronous data being exchanged at the beginning of each window and with asynchronous data exchanged during the remaining portion of each window. Virtual circuits utilizing the synchronous packets can deliver telephone conversations without degradation in voice quality and yet the system can also transmit asynchronous data packets.

2 Claims, 5 Drawing Sheets



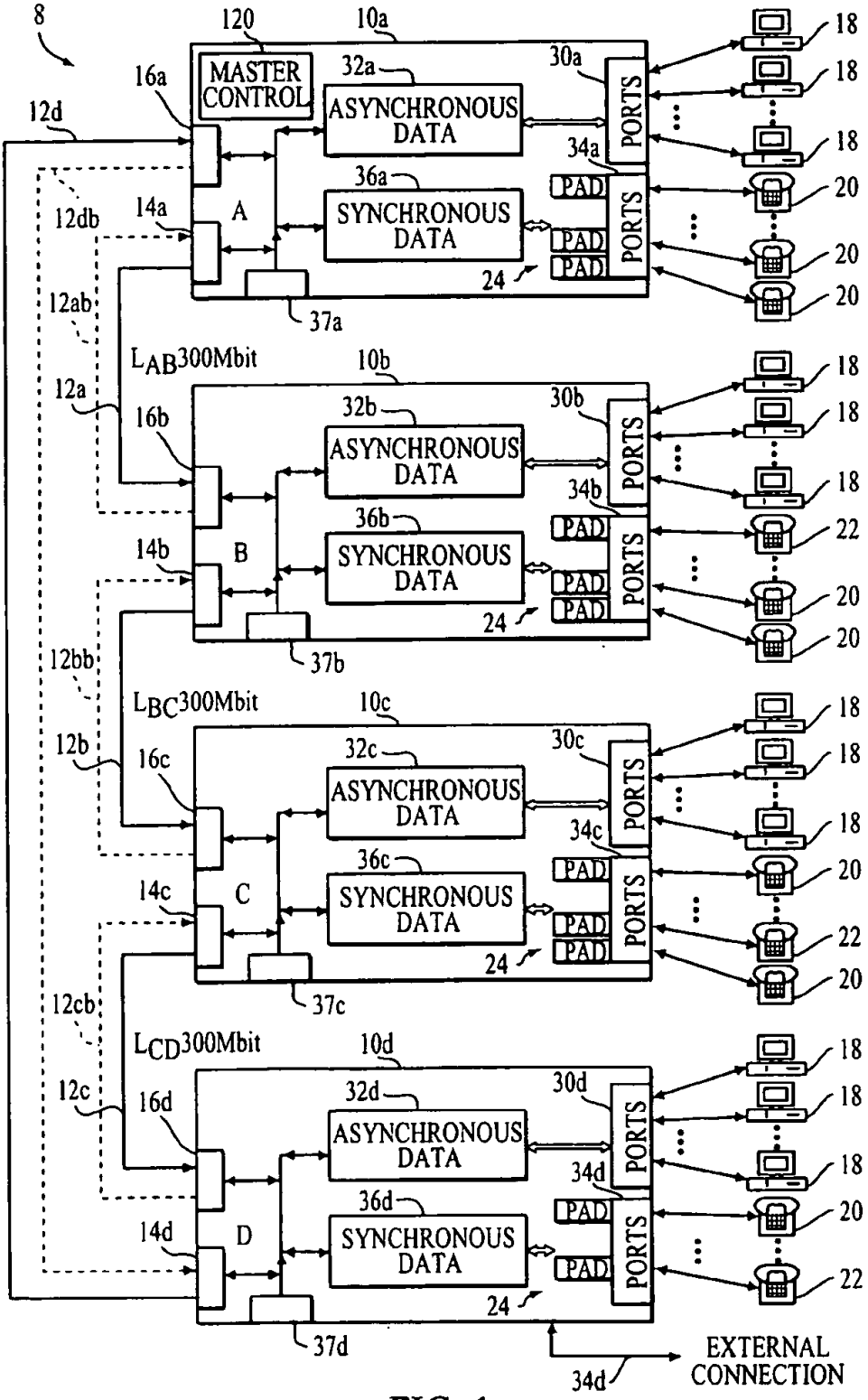

8
120
10a
MASTER CONTROL
16a
12d
32a
ASYNCHRONOUS DATA
30a
PORTS
34a
PAD
36a
SYNCHRONOUS DATA
12db
14a
A
12ab
24
37a
18
20
L_{AB}300Mbit
10b
12a
16b
32b
ASYNCHRONOUS DATA
30b
PORTS
34b
PAD
36b
SYNCHRONOUS DATA
14b
B
12bb
24
37b
22
L_{BC}300Mbit
10c
12b
16c
32c
ASYNCHRONOUS DATA
30c
PORTS
34c
PAD
36c
SYNCHRONOUS DATA
14c
C
12cb
24
37c
L_{CD}300Mbit
10d
12c
16d
32d
ASYNCHRONOUS DATA
30d
PORTS
34d
PAD
36d
SYNCHRONOUS DATA
14d
D
24
37d
EXTERNAL CONNECTION
34d

FIG. 1

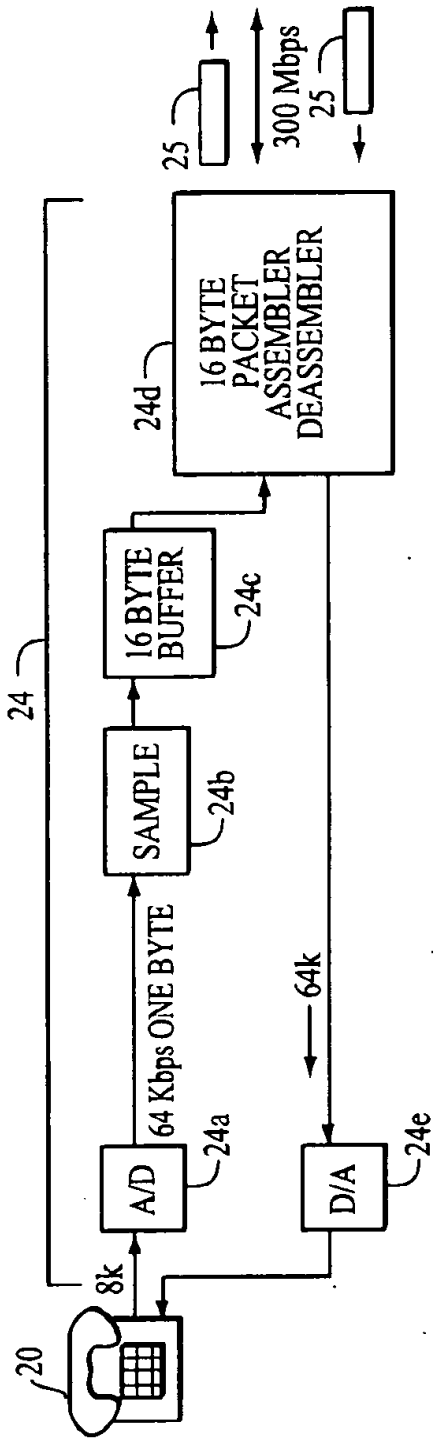

25
300 Mbps
25
16 BYTE PACKET ASSEMBLER DEASSEMBLER
24d
16 BYTE BUFFER
24c
SAMPLE
24b
24
64 Kbps ONE BYTE
64k
A/D
24a
D/A
24e
8k
20

FIG. 2

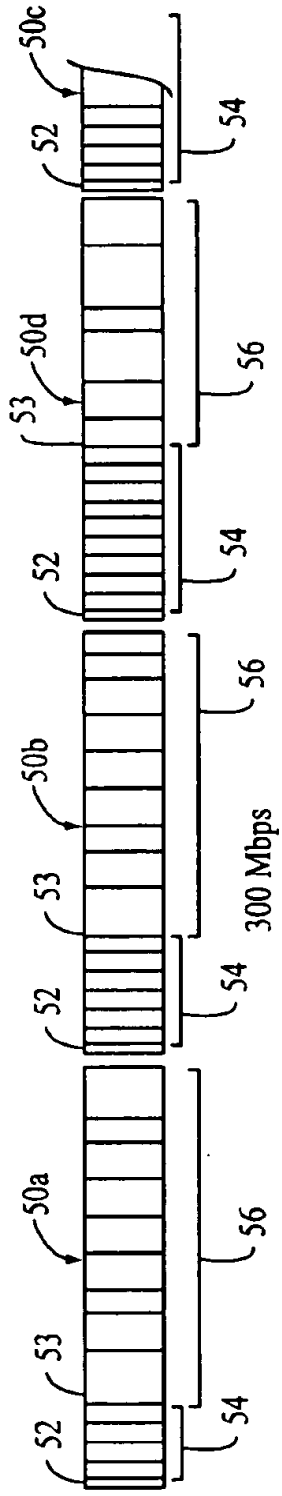

50c
52
54
50d
53
56
52
54
50b
53
56
52
54
300 Mbps
50a
53
56
52
54

FIG. 3

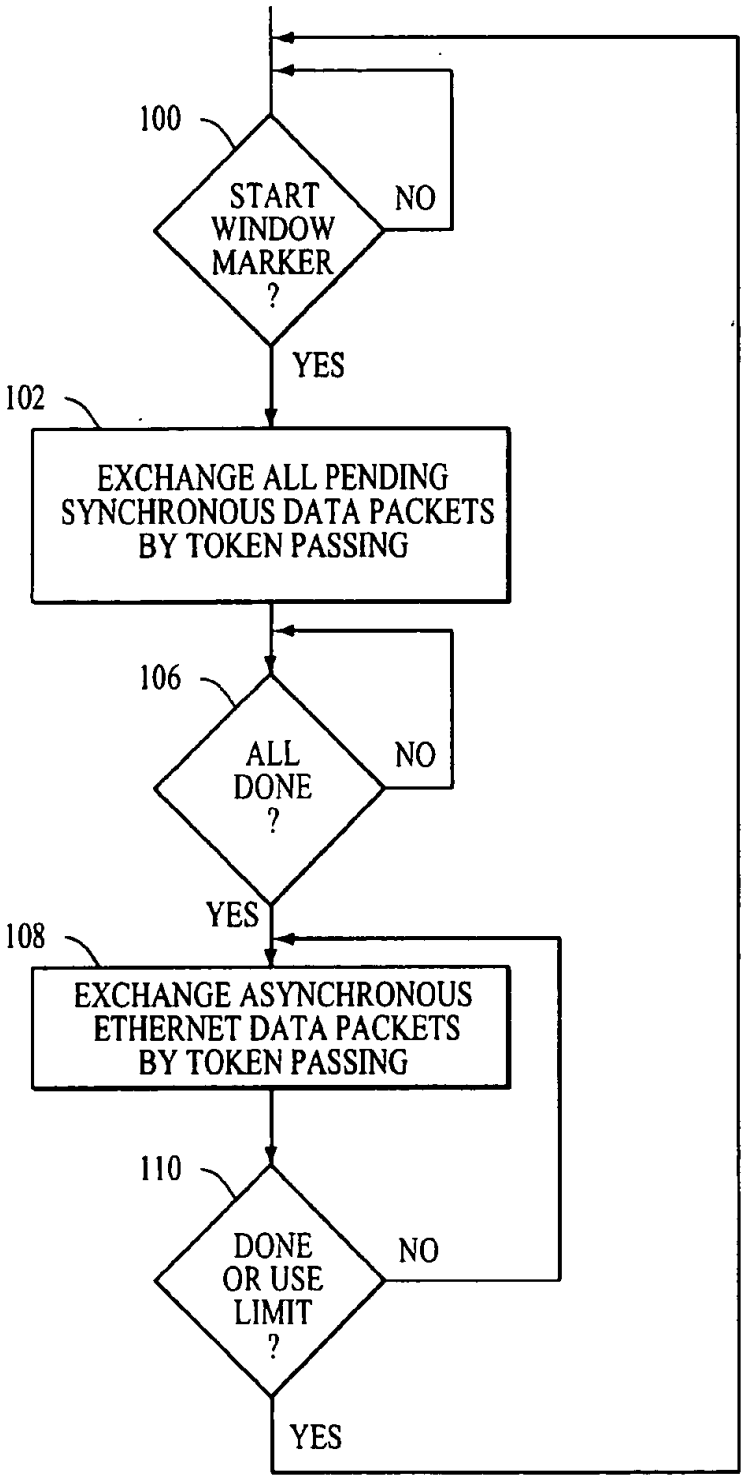

100
START WINDOW MARKER ?
NO
YES
102
EXCHANGE ALL PENDING SYNCHRONOUS DATA PACKETS BY TOKEN PASSING
106
ALL DONE ?
NO
YES
108
EXCHANGE ASYNCHRONOUS ETHERNET DATA PACKETS BY TOKEN PASSING
110
DONE OR USE LIMIT ?
NO
YES

FIG. 4

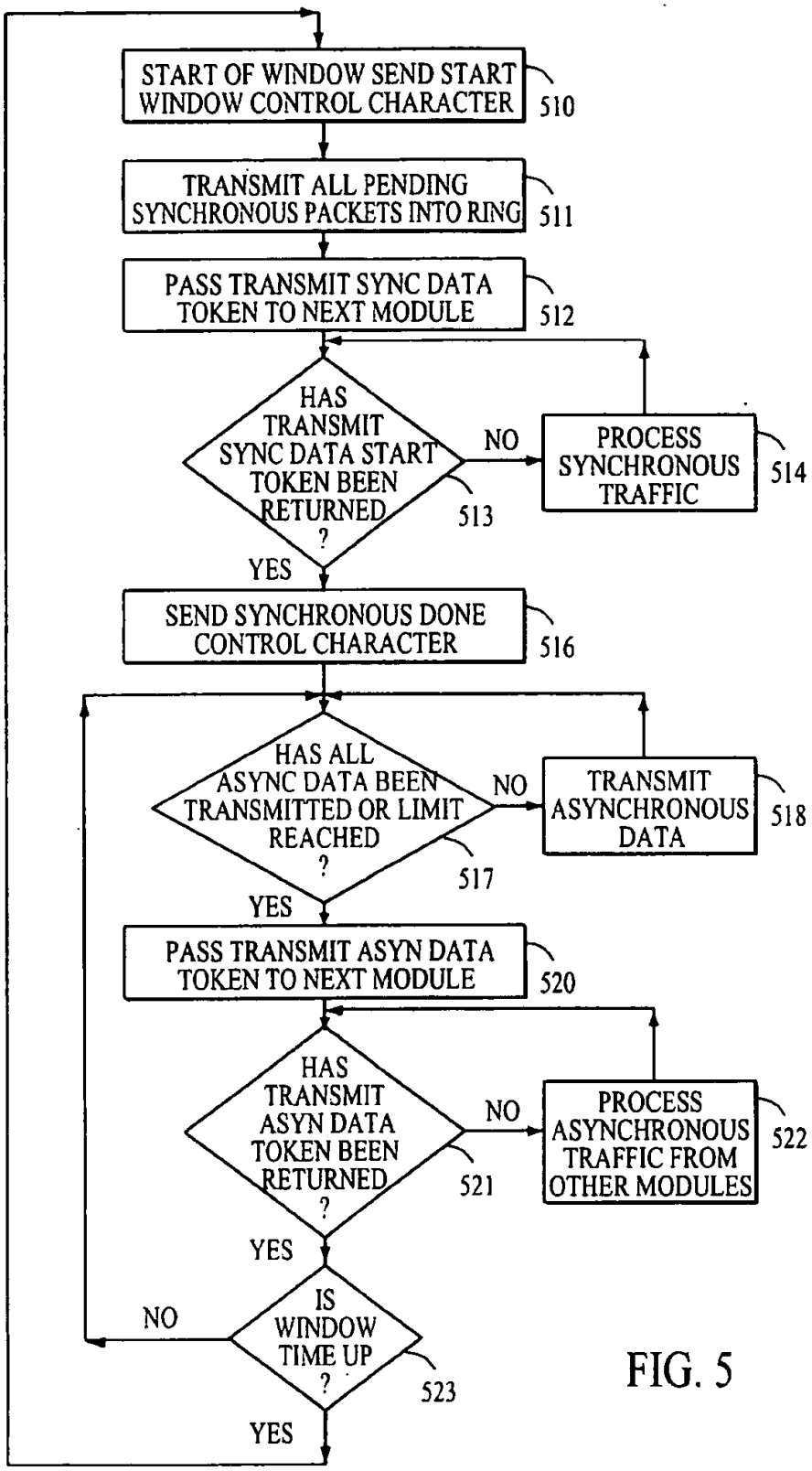
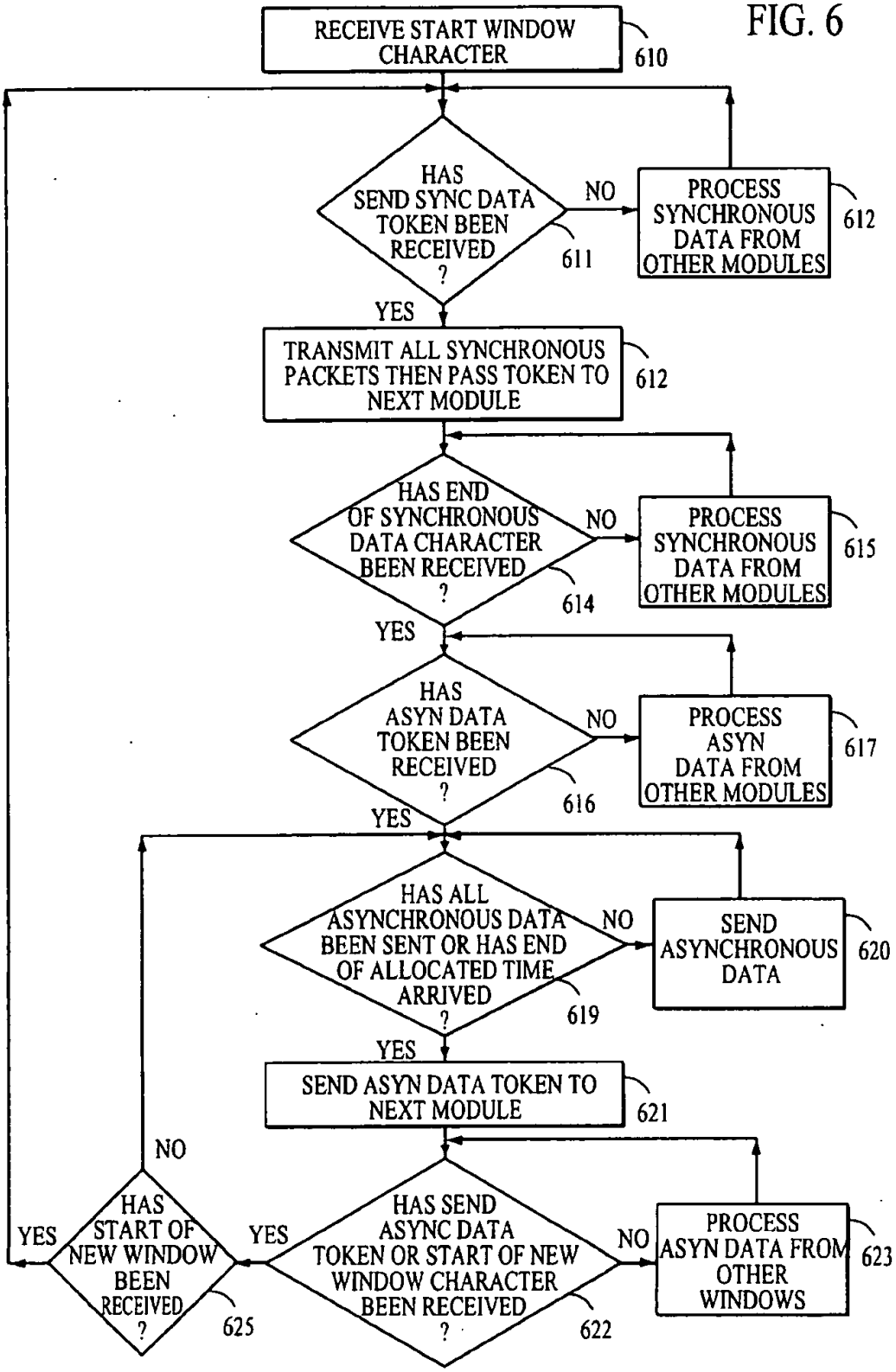

START OF WINDOW SEND START WINDOW CONTROL CHARACTER
510
TRANSMIT ALL PENDING SYNCHRONOUS PACKETS INTO RING
511
PASS TRANSMIT SYNC DATA TOKEN TO NEXT MODULE
512
HAS TRANSMIT SYNC DATA START TOKEN BEEN RETURNED ?
513
NO
PROCESS SYNCHRONOUS TRAFFIC
514
YES
SEND SYNCHRONOUS DONE CONTROL CHARACTER
516
HAS ALL ASYNC DATA BEEN TRANSMITTED OR LIMIT REACHED ?
517
NO
TRANSMIT ASYNCHRONOUS DATA
518
YES
PASS TRANSMIT ASYN DATA TOKEN TO NEXT MODULE
520
HAS TRANSMIT ASYN DATA TOKEN BEEN RETURNED ?
521
NO
PROCESS ASYNCHRONOUS TRAFFIC FROM OTHER MODULES
522
YES
IS WINDOW TIME UP ?
523
NO
YES

FIG. 5

FIG. 6


RECEIVE START WINDOW CHARACTER
610
HAS SEND SYNC DATA TOKEN BEEN RECEIVED ?
611
NO
PROCESS SYNCHRONOUS DATA FROM OTHER MODULES
612
YES
TRANSMIT ALL SYNCHRONOUS PACKETS THEN PASS TOKEN TO NEXT MODULE
612
HAS END OF SYNCHRONOUS DATA CHARACTER BEEN RECEIVED ?
614
NO
PROCESS SYNCHRONOUS DATA FROM OTHER MODULES
615
YES
HAS ASYN DATA TOKEN BEEN RECEIVED ?
616
NO
PROCESS ASYN DATA FROM OTHER MODULES
617
YES
HAS ALL ASYNCHRONOUS DATA BEEN SENT OR HAS END OF ALLOCATED TIME ARRIVED ?
619
NO
SEND ASYNCHRONOUS DATA
620
YES
SEND ASYN DATA TOKEN TO NEXT MODULE
621
HAS SEND ASYNC DATA TOKEN OR START OF NEW WINDOW CHARACTER BEEN RECEIVED ?
622
NO
PROCESS ASYN DATA FROM OTHER WINDOWS
623
YES
HAS START OF NEW WINDOW BEEN RECEIVED ?
625
NO
YES

COMBINED SYNCHRONOUS AND ASYNCHRONOUS MESSAGE TRANSMISSION

RELATED APPLICATION

The present invention is a continuation-in-part of application Ser. No. 60/098,297 filed Aug. 27, 1998 entitled "Combined Synchronous and Asynchronous Message Transmission."

BACKGROUND OF THE INVENTION

Packet switching systems transmit data by breaking the data into relatively small manageable pieces called packets. Packet switching can be used to transmit data in both computer networks and in telephone voice networks. Telephone packet switching networks transmit a series of packets over the same route in the network. Such systems in effect establish a virtual circuit from the point where a series of packets enters the network to the point where the packets are delivered. Packet switching networks establish virtual circuits through the network in order to transmit voice without delay and distortion.

Protocols such as the Internet ITPC protocol can transmit voice without establishing a virtual circuit connection, however, voice transmission using this type of protocol generally has less quality than voice transmitted using protocols which establish virtual circuits between the input point and the output point in the network.

Today, some voice transmissions are being made over packet protocols (such as the Internet) which do not establish virtual circuits. Voice connections over such circuits are of relatively low quality. The packet protocols which are used in the public telephone network are packet protocols which establish virtual circuits and which transmit all the packets that constitute a conversation over the same route through the network. Thus they provide high quality connections.

Data communication protocols can be characterized as either synchronous or asynchronous. Examples of widely used synchronous protocols are the X.25 protocol, and the frame relay protocol. Examples of widely used asynchronous protocols are the Ethernet, FDDI and ATM protocols. The X.25 protocol, the frame relay protocol and the ATM protocol are widely used in telephone systems. The Ethernet protocol and the FDDI ring protocol are widely used in local area networks (LANS) and wide area networks (WANS) that are used to interconnect computer systems.

There are various well known techniques for controlling asynchronous networks. One technique termed "carrier sense, multiple access with collision detection (CSMA/CD) is used in Ethernet networks. Another technique called token passing is used in FDDI ring networks.

Explanations of various synchronous and asynchronous protocols, and an explanation of CSMA/CD and FDDI ring networks is for example given in a book entitled "Voice and Data Communications Handbook" by Regis J. Bates and Donald Gregory which is published by McGraw Hill.

SUMMARY OF THE INVENTION

The present invention provides a ring protocol and system that combines synchronous and asynchronous transmission techniques. The ring can interconnect a number of modules and be utilized to transmit both fixed and variable packets between the modules. Communication time is broken into a sequence of fixed length windows. At the beginning of each window the modules communicate using a synchronous protocol. That is, at the beginning of each window, if any unit has synchronous traffic, such traffic is transmitted using a synchronous ring protocol and fixed length packets. Virtual circuits can be established between the modules using the synchronous fixed length packets communicated at the beginning of each window. When it is desired to establish a virtual circuit between any of the modules in the ring, each module is assured that at the beginning of each window, space will be allocated to transmit a synchronous fixed length packet to another module in the ring. The windows occur frequently enough that a virtual voice grade circuit can be established between the modules. After all synchronous packets required during any window have been transmitted, asynchronous variable length data packets are transmitted around the ring. Limits are provided relative to the number of asynchronous packets any one module can transmit, thereby avoiding monopolization of the ring by any one module. The modules are synchronized by a periodically circulating a timing control character around the ring.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 is an overall block diagram of the preferred embodiment.

FIG. 2 is a diagram of the package assembler and disassembler (PAD) portion of the modules for coupling to analog telephone devices.

FIG. 3 shows a repeating sequence of fixed-length time windows used in allocating data traffic between the modules of FIG. 1.

FIG. 4 is a system-level flow chart illustrating overall operation of the communication system of FIG. 1.

FIG. 5 is a program flow chart showing the operation of the module which executes certain control among the modules of FIG. 1.

FIG. 6 is a flow chart showing the operation of modules of FIG. 1 when receiving and processing information exchanged there-between.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

A preferred embodiment of the invention is shown in FIG. 1 and described herein. FIG. 1 shows a communication system 8 which supports integrated exchange of asynchronous and synchronous data between a number of modules 10*a*, 10*b*, etc. The data exchanged between the modules includes data traffic from and to computers 18 and voice traffic from conventional telephone devices 20 and 22. The data traffic from and to the computers 18 can include all of the various types of data traffic conventionally generated by computers such as video data, pc-phone data, etc. The system shown in FIG. 1 establishes "virtual circuits" between modules 10 for telephone traffic (i.e. synchronous transfer) and to also manages exchange of asynchronous information transfer between modules 10.

As shown in FIG. 1 system 8 includes a collection of modules 10 organized in a ring architecture. Information travels from one module 10 to a successive module 10 as required. The information on the ring can be divided into three categories, namely, data packets, tokens, and control characters (including a timing character). As used herein the following terms have the following meaning.

A "byte" consists of ten bits. Eight data bits are coded into a ten bit byte. Since a ten bit byte is used to encode 8 bits of data, a byte can be decoded into 256 different data words

plus 768 additional decodes. Some of the additional decodes are used to form control characters, a timing character, and tokens. Other ones of the additional decodes are used for purposes unrelated to the present invention. Such decoding is conventional.

A "control character" consists one byte. The specific byte that forms each control character is selected from the decodes which do not form data words. There are three control characters which are used to implement the present invention. One control character indicates the start of a window, a second indicates the end of synchronous data transfer, and the third indicates the end of a window. When a module receives a control character it immediately retransmits the character to the next module in the ring.

A "token" consists of two bytes of data. As with control characters, the specific bytes that form each token are selected from the decodes which are not otherwise assigned. When a module receives a token, it only retransmits the token to the next module if certain conditions have been met. There are two tokens used in the implementation of the present invention. One token indicates that a module should begin transmitting its synchronous data and the second indicates that a module should begin transmitting asynchronous data. A module only passes a token to the next module after a module that receives a token has completed the task initiated by the token.

A "timing character" consists of one byte. This one byte is selected from the decodes not otherwise assigned.

A "synchronous packet" consists of 16 bytes of data plus three bytes of address information.

A "asynchronous packet" consists of 64 to 1524 bytes of data plus an 8 byte Ethernet header.

As each module **10** receives bytes (i.e. information packets, control characters, and tokens) from its predecessor in the ring, the module copies the bytes it receives internally and retransmits the bytes to a successive module **10** if appropriate. Bytes thereby flow at high speed around the ring architecture from one module **10** to another module **10**. It is noted that tokens are only transmitted from one module to another module when a task initiated by the token has been completed.

The ring architecture allows use of a variable number of modules **10**. Four such modules **10**, individually **10***a*–**10***d*, are shown in the particular embodiment described herein. It will be understood, however, that more modules **10** may be inserted into the ring architecture or that some of modules **10***a*–**10***d* may be removed from the ring architecture. Thus, modules **10** "stack" to meet use requirements, e.g., system **8** expands, to follow a growing user population or capacity requirement.

Each module **10** communicates with two adjacent modules through interconnecting communication links **12**. Each of the links **12** is bi-directional. For example link **12***a* goes from module **10***a* to **10***b* and link **12***ab* goes from module **10***b* to module **10***a*. In normal operation the system uses links **12***a*, **12***b*,**12***c*, and **12***d*. If one of these links is down (i.e. broken) the system automatically switches to links **12***ab* to **12***da*. Such use of a set of backup links is conventional.

Link **12***a* couples the output port **14***a* with the input port **16***b* of module **10***b*. Link **12***b* couples the output port **14***b* with the input port **16***c* of module **10***c*. Similarly, link **12***c* couples the output port **14***c* with the input port **16***d* of module **10***d*. Finally, link **12***d* couples the output port **14***d* with the input port **16***a* of module **10***a*. Each communication link **12** is a high-speed communication path. The capacity for links **12** is established depending on the number of modules **10** involved and the number of user devices attached to modules **10**. In the specific embodiment shown herein communication links **12** operate at 300 Mbps.

Modules **10** handle both synchronous data packets and asynchronous data packets. The synchronous packets **25** are fixed-length 16 byte packets **25**. The asynchronous packets are variable length packets **35**. Each module **10** has a number of asynchronous ports **30** (designated **30***a* to **30***d*) coupled to an asynchronous data buffer **32** and a number of synchronous ports **34** (designated **34***a* to **34***d*) coupled to a synchronous data buffer **36**. Computers **18** are connected to asynchronous data ports **30** and telephones **20** and **22** are connected to synchronous data ports **34**.

Modules **10** are interconnected by links **12**. Links **12** establish a combined synchronous and asynchronous message transmission data network whereby computers **18** may share resources and data and whereby telephone conversations may be conducted among the population of telephones **20** and **22**.

Each module **10** has a timer **37** (individually identified as timers **37***a* to **37***d*) which controls the timing within the module. Each module also includes a conventional programmed RISC processor and an associated memory which store and execute the programming operations described below.

Each module also has a plurality of user devices connected thereto. As shown in FIG. **1**, the user devices connected to the modules **10** include various computer work stations or terminals **18**, analog telephones **20**, and digital phones **22**.

Each analog telephone **20** is connected to a packet assembler and disassembler (PAD) **24**. FIG. **2** illustrates in more detail a PAD **24**. A PAD includes an analog-to-digital converter **24***a*, a sampling circuit **24***b*, a packet buffer **24***c*, and a packet assembler and disassembler block **24***d*. Each PAD **24** produces a sequence of 16 byte packets **25** representing voice sampled from a corresponding analog telephone **20**. Such a sequence of packets carry "one side" of a telephone conversation.

Each PAD **24** also receives a sequence of 16 byte packets **25** for audible presentation of voice at the corresponding analog telephone **20**. Block **24***d* drives a digital-to-analog converter **24***e* with incoming packets **25**, i.e., the "other side" of a telephone conversation involving a user and telephone **20**. Thus, block **24***d* operates within a given module **10** providing and receiving packets **25** representing an analog telephone conversation and the associated normal in-band telephone signals. Packet **25** transport occurs at 64 kbps in order to support the full duplex telephone traffic.

Digital telephones **22** produce similar packets **25** representing one side of a telephone conversation, i.e., voice sampled by digital telephone **22**, and also receive a sequence of packets **25** for audible presentation at a digital telephone **22**. Digital telephones **22** exchange packets **25** with a module **10** at sufficient speed to support a full duplex telephone conversation, i.e., 64 kbps.

Telephone conversation data from telephones **20** and **22** is packetized in the fixed-length 16 byte packets **25**. The packets containing voice data must be delivered in a timely manner in order to maintain acceptable quality of voice communication. In order to accomplish timely delivery of data representing voice communications, all such data is handled by the present invention in a synchronous fashion. Since such data is handled in synchronous fashion conventional "virtual circuits" can be established between user devices, e.g., between members of the population of tele-

phones 20 and 22. High quality telephone connections can therefore be achieved. There is no perceptible degradation in voice quality, because no more than about a four and a half millisecond delay exists in delivery of any given packet 25 from end terminal to end terminal (i.e. from telephone to telephone).

Computer work stations 18 produce data for delivery to other stations 18 and receive data from other stations 18. Because variation in delay and variation in packet size is generally acceptably in communications between stations 18, such data is managed in an asynchronous fashion when transported via modules 10. Information exchanged among stations 18 is divided into "Ethernet" type packets, i.e., variable sized packets including addressing information according to an Ethernet type addressing schemes.

The time frame for communication on links 12 is divided into a sequence of windows. FIG. 3 illustrates a sequence of windows 50, individually identified as windows 50*a*, 50*b*, etc. Each window 50 is two millisecond long

Each window 50 begins with a "start window" control character or field 52 (which is one byte long). The start window control character indicates the onset of a window 50. The remainder of each window 50 is dedicated first to all pending synchronous data transmissions and then to asynchronous data transmissions. More particularly, a first portion 54 of each window 50 is dedicated to exchange of all pending synchronous data packets 25. After all pending synchronous data packets 25 have been exchanged among modules 10, a second control character (not explicitly shown in FIG. 3) is transmitted around the ring to indicate the end of the synchronous transmissions. A second portion 56 of each window 50 is dedicated to exchange of asynchronous data packets 35. At the end of each window another control character (not explicitly shown in FIG. 3) is transmitted around the ring. As will be explained later, tokens and timing characters are also transmitted around the ring.

The length of window 50 is two milliseconds long. The length of windows 50 is established by taking into account the bandwidth of the various communication paths. The length of window 50 is established so that all synchronous data can be delivered during each window and so that after the synchronous data is transmitted, sufficient reserve will remain in each window 50 to conduct exchange of asynchronous data. The actual allocation of a given window 50 between synchronous and asynchronous data is dynamic. The allocation depends on the amount of pending synchronous data packets 25 which must be transmitted during the first portion of a given window 50. As the number of telephone conversations increases, the portion 54 of window 50 used for such conversations increases.

Thus the allocation between synchronous and asynchronous data in a given window 50 is not fixed but rather a function of the amount of synchronous data pending at the beginning of the window 50 with the remaining portion 56 being used for asynchronous data. A control character 53 which indicates that synchronous traffic is "all done" separates portions 54 and 56 of each window. This control character indicates the end of synchronous data transmission and the beginning of asynchronous data transmission within a given window 50.

FIG. 4 illustrates, at a system level, data exchange during a given window 50. As shown in FIG. 4, processing loops at decision block 100 until start window control character 52 appears on links 12. Processing then advances to block 102 where modules 10 exchange all pending synchronous data packets, i.e., deliver all pending packets 25 in the synchronous data buffers 36. Block 102 represents the overall exchange of all pending synchronous data packets 25 among modules 10*a*–10*d* by ring message exchange.

As indicated by decision block 106 a determination is made that all modules 10 have completed exchange of pending synchronous data, e.g., all modules 10 have delivered all pending synchronous data packets 25. In other words, all synchronous data in buffers 36 at the onset of the current window 50 have been transmitted via links 12 to an appropriate module 10. At this point, communication among modules 10 switches from a synchronous mode of operation to an asynchronous mode of operation allowing variable length packets and an alternate addressing scheme. Asynchronous transmission is done using Ethernet packet rules and addressing codes to route the variable length packets to particular modules 10 and to corresponding user devices attached thereto. Ethernet packaging rules allow packets of varying length between 64 and 1524 bytes

To prevent monopolization of window 50 by one module, each module 10 limits its use of portion 56 of each window so as to allow other modules 10 to deliver asynchronous data. Thus, block 108 represents delivery of a limited amount of asynchronous packets followed in decision block 110 which tests use limitations. During block 108, a module 10 sends a certain number of Ethernet packets to a successive one of modules 10. Such module 10 limits its further use of the asynchronous portion 56 of a given window 50. That is, each module 10 is allowed a limited number of asynchronous data bytes per given window 50. In the embodiment described herein, each module 10 transmits a maximum of 4000 bytes of asynchronous data in any given window 50. Thus, system level operation loops at blocks 108 and 110 until all modules 10 have reached their use limit for the current window 50 or have delivered all pending asynchronous data. Processing eventually returns to block 100 where system 8 waits for occurrence of the start window control character 52 and a next window 50.

The modules 10 in general operate on a "peer" basis. However, one of modules 10 is given some degree of control over the process. In the embodiment shown, module 10*a* executes master control, that is, to some extent module 10*a* orchestrates the exchange of information on links 12 and it is in effect a timing master for the system. Modules 10 other than module 10*a* may be inserted and removed from the system as needed or desired without corrupting an overall control strategy, However, there must always be a control module 10*a*.

Master control module 10*a* makes use of control characters and tokens to orchestrate packetized information exchange within system 8. Modules detect the receipt of a token or control character by detecting one of the decodes of a byte other than the 256 data decodes. When a module 10 receives a control character or a timing character, it immediately retransmits the control character or timing character to the next module 10. When a module 10 receives a token, the token is held until the module 10 is ready to retransmit the token, i.e. until the module is ready to relinquish its right to send packets of a particular type.

FIG. 5 illustrates programming with respect to operation of module 10*a*. Module 10*a* is the master control module. As indicated by block 510, the process begins when module 10*a* transmits a "window start" control character 52 (see FIG. 3). Control character 52 is sent immediately around the ring architecture because when a module 10 receives a control character it immediately re-transmit (i.e. repeats) the control character. At this point, all modules 10 are prepared for the

onset of a window **50**. As indicated by block **511** module **10***a* transmits all its pending synchronous packets **25**, i.e., module **10***a* empties synchronous data buffer **36***a*, into the ring on communication link **12***a*. Once module **10***a* has transmitted all of its synchronous data packets **25**, as indicated by block **512** module **10***a* passes the "transmit synchronous packets" token to the next module, i.e., to module **10***b*.

After passing the transmit synchronous packets token to the next module on the ring, module **10***a* processes synchronous packet traffic from other modules until the transmit synchronous packets token is returned to module **10***a*. This is indicated by blocks **513** and **514**. During this time, the remaining modules **10** will each in turn have opportunity to send all synchronous data packets **25** which were pending at the onset of the current window **50**. That is, after module **10***b* receives the token, it transmits all its pending synchronous data packets **25**, i.e., empties synchronous data buffer **36***b*, onto link **12***b*. Module **10***b* then passes the token to module **10***c*, giving module **10***c* an opportunity to send all its synchronous data packets **25** onto link **12***c*. The token is then passed to module **10***d*. When module **10***d* receives the token, it in turn submits all its synchronous data packets **25** which were pending at the onset of the current window **50** onto link **12***d*. As each module **10** submits its synchronous data packets **25** onto the ring architecture, each packet **25** reaches a target or module **10** which has attached thereto one of telephones **20** or **22** so as to complete a virtual circuit.

Eventually, all modules **10** will have had an opportunity to submit synchronous data packets **25** onto the ring and the "transmit synchronous packets" token will return to module **10***a*. Processing then advances from decision block **513** to block **516** and module **10***a* sends the "all done synchronous data" control character **53** out on link **12***a*. Each of modules **10***b*–**10***d* thereby receive the "all done synchronous data" control character **53** indicating a transition from synchronous data exchange to asynchronous data exchange. As indicated by blocks **517** and **518**, module **10***a* transmits asynchronous data packets **35** onto link **12***a*. As may be appreciated, each of the asynchronous data packets pass around the ring and eventually reach the intended destination, i.e., one of modules **10***b*–**10***d* addressed as the destination address in the packet.

Module **10***a* stops sending asynchronous data packets when one of two conditions is satisfied as indicated by decision block **517**. Transmission of asynchronous packets by module **10***a* stops when module **10***a* determines that it has no more asynchronous data to transmit, (i.e., asynchronous data buffer **32***a* is empty) or if module **10***a* has reached its use limit. In defined a module is limited to transmitting 4000 bytes in one session. If module **10***a* has not reached its use limit and if there are additional asynchronous data packets **35** in buffer **32***a*, then processing returns to block **518** and module **10***a* continues to transmit asynchronous data packets Eventually, module **10***a* either reaches its use limit or exhausts pending asynchronous data in buffer **32***a*. Processing then advances to block **520** and module **10***a* passes the token to the next module, i.e., to module **10***b*.

After passing the token to the next module, module **10***a* will process asynchronous data traffic as indicated by blocks **521** and **522**. Module **10***a* checks for return of the token as indicated by decision block **421**. That is processing as indicated by blocks **521** and **522** continues until the token is returned to module **10***a*. When the token returns to module **10***a*, all modules **10** have had opportunity to send asynchronous data packets **35** onto the ring at least once up to their given limit, i.e., at least 4000 bytes.

At this point, some portion of window **50** may remain. This is determined as indicated by block **523**. If more time remains module **10***a* can take advantage of this opportunity to send more asynchronous data packets **35**. As indicated in FIG. **5** The "no" output from block **523** goes back to block **517** and the process repeats.

Eventually, module **10***a* runs out of time in the current window **50** for transmission of additional packets **35**. An end of ring control character is then transmitted around the ring. Processing then returns to block **510** where module **10***a* again sends control character **52** and the process repeats.

Module **10***a* also provides a timing reference for the system. Module **10***a* includes an interval timer **37***a* which produces interrupt signal every 15.625 microseconds. This timing reference signal is transmitted from module **10***a* to the other modules in the ring. When the timing interrupt occurs, module **10***a* transmits the special timing control character. The timing control character is inserted into any packet that is being transmitted at the time the interrupt occurs. Thus, some synchronous packets **25** may be 17 bytes long after the clock control character is inserted therein. The additional delay introduced, i.e., a 16 byte synchronous packet **25** versus a 17 byte synchronous packet **25**, does not introduce any noticeable delay to persons engaged in a conversation. The timing character is a 10 bit character which is not used for any other purpose and which each units recognizes as a timing character. When a unit on the ring (other than module **10***a*) detects this character, it repeats the character to the next unit on the ring and at the same time it re-synchronizes its internal clock **37**. That is, the clock **37** in each unit is re-synchronized when the timing character is detected. In this way the clocks **37** in the various modules are kept in close synchronization. It is noted that as shown herein it is the control module **10***a* which introduces the timing character onto the ring, any one of the modules could perform this function.

FIG. **6** illustrates the programming for modules **10** other than module **10***a*. (FIG. **5** shows the programming for module **10***a*). That is, FIG. **6** illustrates programming for modules **10***b*–**10***d*. The process starts as indicated by block **610** when a module receives a "start window" control character **52**. After receiving a "start window" control character a module looks for a token with indicates that the module should start transmitting synchronous data. Processing continues iteration between blocks **611** and **612** until the module **10** receives the "start transmitting synchronous data" token. As indicated by block **611**, when a module receives the "start transmitting synchronous data" token the module begins transmission of its synchronous data. Once a module **10** has transmitted all pending synchronous data packets **25** into the ring, it passes the "start transmitting synchronous data" token to the next module **10**. This gives the next module **10** opportunity to submit its synchronous data packets **25**.

Once a module **10** has passed the "start transmitting synchronous data" the module processes synchronous packet traffic from other modules as indicated by block **615**. Decision block **614** indicates that a module looks for "all synchronous traffic done" control character **53**. Until the "all synchronous traffic done" control character **53** appears, processing iterates at blocks **614** and **615** and the module processes any synchronous data packets **25** appearing in the ring architecture from other modules. When the "all synchronous traffic done" control character **53** does appear processing advances to block **616** and **617** where the module processes any asynchronous packet traffic appearing in the ring architecture from other modules. In decision block **616**, module **10** looks for the "send asynchronous data" token. Processing iterates at blocks **616** and **617** until the module

receives the "send asynchronous data" token. "send asynchronous data" token is received asynchronous data packet **35** are transmitted as indicted by block **620**. After transmitting each asynchronous data packet **35**, the module determines as indicated by block **619** whether it has any additional asynchronous data packets to transmit, or if the module has reached its use limit, e.g., has transmitted 4000 bytes of asynchronous information in the current window **50**. Processing iterates at blocks **619** and **620** until no further asynchronous data packets remain or until the module **10** has reached its allotment or use limit allowed in the current window **50**. Processing then advances to block **621** and the module passes the "send asynchronous data" token to the next module **10**.

After each of the modules **10** has had an opportunity to transmit both synchronous and asynchronous traffic during a particular window, there may still be time remaining in the window. When this condition occurs, the modules are given another chance to transmit additional asynchronous packets. This condition is illustrated in FIG. **6** by the path from block **622** through block **625** to the entry of block **619**.

Following block **621**, processing iterates between blocks **622** and **623** where the module processes any further asynchronous packet traffic from other modules and looks for the occurrence of the end of window and the start new window control characters. If an end of window and start of new window control characters have not been received the processing goes from block **622** to **625** to **619**. When and end of window and start of new window control characters are received the processing returns to block **611**.

As an example of the capacity of the system, it is noted that a voice switching capacity on the order of 256 simultaneous, full duplex calls may be implemented on a stack of eight modules **10**. Each full duplex voice call consumes 64 kbps of data bandwidth. This translates into: (64,000*2)*256=32,768,000 or 32 Megabits (Mbps) of voice switching bandwidth on links **12** to support 256 simultaneous full duplex voice calls. This is exclusive of framing overhead, which will be dependent on the hardware implementation.

There are also other capacity considerations. Services which use 'redirection', e.g., voice compression, voice recognition or fax services, all are very processor bandwidth intensive. This component creates an 'overhead factor' on the base voice switching bandwidth independent of framing overhead. Using an estimated overhead factor of thirty-three percent, the voice switching bandwidth requirement increases from about 32 Mbps to about 44 Mbps.

For asynchronous data capacity, a minimum carrying requirement of a single 100 Mbps Ethernet will meet a given level of computer network use expectations. This brings the aggregate carrying capacity for system **8** to about 144 Mbps.

For control and management an overhead of not more than about 4 Mbps per module **10** is expected. Of this, 1 Mbps is reserved for true inter-module **10** communication, 0.5 Mbps is reserved for network management, 1.5 Mbps is reserved for call accounting and 1 Mbps is reserved for event logging and tracing. For eight modules **10** in a system **8**, a subtotal of (8*4,000,000) or about 32 Mbps. This brings the entire switching capacity requirement for system **8** to about 176 Mbps. Establishing a 300 Mbps capacity for links **12** supports this expected switching capacity.

Naturally, other systems could be implemented using the present invention with different requirements and capacity considerations. For example various numbers of modules **10** could be connected in a ring utilizing the present invention, While the invention is described as applied to a LAN environment, it is noted that the invention could also be applied in a WAN environment. It is also noted that the communication path between modules **10** could be either electrical or optical without departing from the present invention.

It should be appreciated that the computer data communicated between modules **10**, could include all the various kinks of data normally transmitted between computer. For example such data could include video data and PC-telephone data.

A combined synchronous and asynchronous message transmission method and apparatus has been shown and described. The integration of synchronous and asynchronous message transmission into a single communication system provides opportunity for integrated communication services incorporating both computer data and voice data. Despite integration of synchronous and asynchronous data, synchronous data arrives in timely fashion without degradation in voice quality.

As described herein data from computer **18** is treated as asynchronous data. However, under certain conditions, computers **18** may be required to deliver time-sensitive information in a synchronous manner, and could be treated as such by a corresponding module **10**.

System **8** also includes an external connection **38**, e.g., a high speed telephone or network connection, whereby other systems may introduce information into system **8** or take information from system **8**. Such links are handled similar to links directly connected to modules **10**.

It is noted herein that a single asynchronous protocol is used. However, additional control schemes may also be employed. For example such protocols could be used to allow further exchange of asynchronous data when modules **10** have reached their use limit for asynchronous data, but window **50** has not yet expired.

It will be appreciated that the present invention is not restricted to the particular embodiment that has been described and illustrated, and that variations may be made therein without departing from the scope of the invention as found in the appended claims and equivalents thereof.

What is claimed is:

1. A communication system comprising:

a plurality of communication modules interconnected to exchange information there between,

said information including a variable amount of synchronous information and a variable amount of asynchronous information,

said modules providing and receiving said synchronous information in substantially fixed length packets, said modules providing and receiving said asynchronous information in variable length packets, each of said packets comprising a plurality of multi bit bytes;

a communication channel coupling said modules together, operation of said communication channel being divided into repeated fixed length windows, said channel operating in accordance with a token passing protocol,

each module transmitting a variable number of fixed length packets between said modules during a first portion of each of said fixed length windows and each module transmitting a variable number of variable length packets between said modules during a second portion of each of said repeated fixed length windows,

each module transmitting a token to the next module when that module has completed the transmission of all

synchronous packets from that module to other modules, in order to indicate that the next module can begin transmission of synchronous packets, and

a timer which periodically inserts a timing byte into the string of bytes being transmitted between said modules on said communication channel, each of said modules including a timer which is resynchronized in accordance with said timing byte.

2. A system for transmitting synchronous and asynchronous information, said system comprising:

a plurality of modules coupled in a ring, each module receiving input from a predecessor module and providing output to a successor module, each of said modules referencing a repeating sequence of time windows, each time window including a first portion dedicated to exchange of synchronous data and a second portion dedicated to exchange of asynchronous data, said ring operating in accordance with to a token passing protocol, said synchronous data being in the form of fixed length packets and said asynchronous data being in the form of variable length packets,

said first portion of said time window terminating when all pending synchronous data has been exchanged among said plurality of modules, each module transmitting a token to the next ring when it is completed transmitting all synchronous data pending in that module, use of said second portion of said time window by each module being limited whereby each of said modules has opportunity during said second portion of said window to transmit some asynchronous data,

each module including a clock, synchronization between said modules being maintained by periodically transmitting a character on said ring which is recognized by each module as a timing character and wherein each modules resynchronizes its clock when said character is received.

* * * * *

US005206857A

United States Patent [19]

Farleigh

[11] Patent Number: 5,206,857

[45] Date of Patent: Apr. 27, 1993

[54] APPARATUS AND METHOD FOR TIMING DISTRIBUTION OVER AN ASYNCHRONOUS RING

[75] Inventor: Scott E. Farleigh, Denver, Colo.

[73] Assignee: AT&T Bell Laboratories, Murray Hill, N.J.

[21] Appl. No.: 692,741

[22] Filed: Apr. 29, 1991

[51] Int. Cl.[5] H04L 12/42; H04L 7/04

[52] U.S. Cl. 370/85.5; 370/85.1; 370/85.15; 370/103; 370/109

[58] Field of Search 370/85.5, 13, 17, 85.15, 370/103, 100.1, 85.1; 375/109

[56] References Cited

U.S. PATENT DOCUMENTS

4,539,679	9/1985	Bux et al.	370/88
4,569,042	2/1986	Larson	370/13
4,761,799	8/1988	Arragon	370/103
4,773,065	9/1988	Kobayashi et al.	370/85.1
4,792,966	12/1988	Ballweg	375/112
4,807,259	2/1989	Yamanaka et al.	375/109
4,815,110	3/1989	Benson et al.	370/103
4,845,709	7/1989	Matsumoto et al.	370/85.5
4,866,704	9/1989	Bergman	370/85.4

OTHER PUBLICATIONS

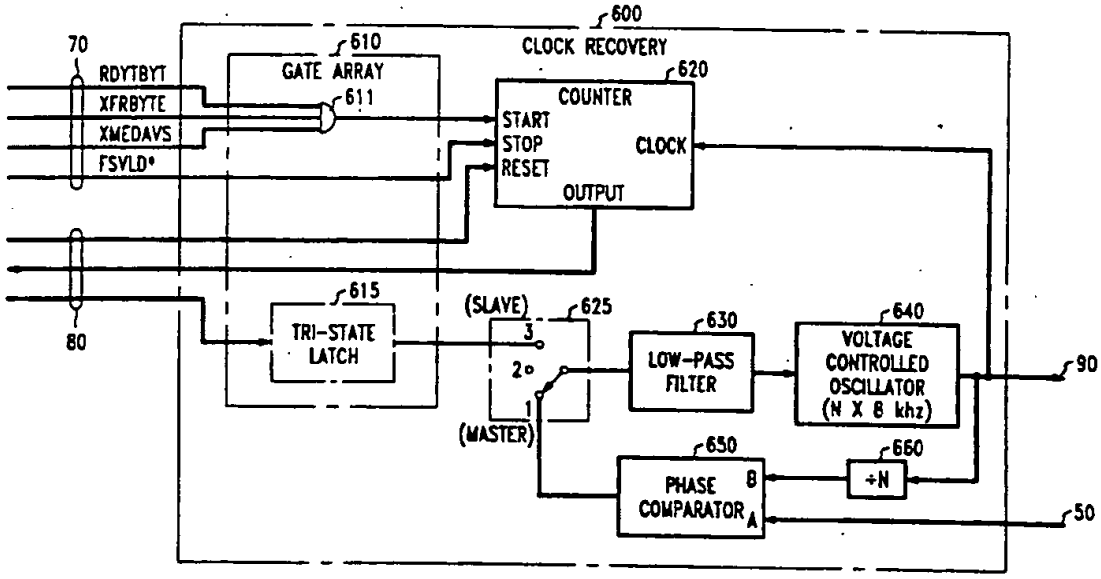
FDDI-A Tutorial, Floyd E. Ross, IEEE Communications Magazine May 1986, vol. 24, No. 5.

Primary Examiner—Douglas W. Olms
Assistant Examiner—Wellington Chin
Attorney, Agent, or Firm—Michael A. Morra

[57] ABSTRACT

An asynchronous, fiber optic, ring network includes a number of nodes where data enters and exits the network. Each of these nodes has its own clock to provide timing needed by equipment at that node. So that synchronous data can be transmitted between predetermined nodes, a master node provides timing information which may be used at any node to synchronize its clock. Timing information comprises the propagation delay around the ring (ring latency) as measured by the master node. This measure of propagation delay is transmitted by the master node as an information packet available at all nodes. Synchronization of any other node with the master requires that the other node perform its own measurement of propagation delay, compare its measurement with that of the master node, and make adjustments to its own clock that tend to decrease the difference between these measurements.

13 Claims, 4 Drawing Sheets



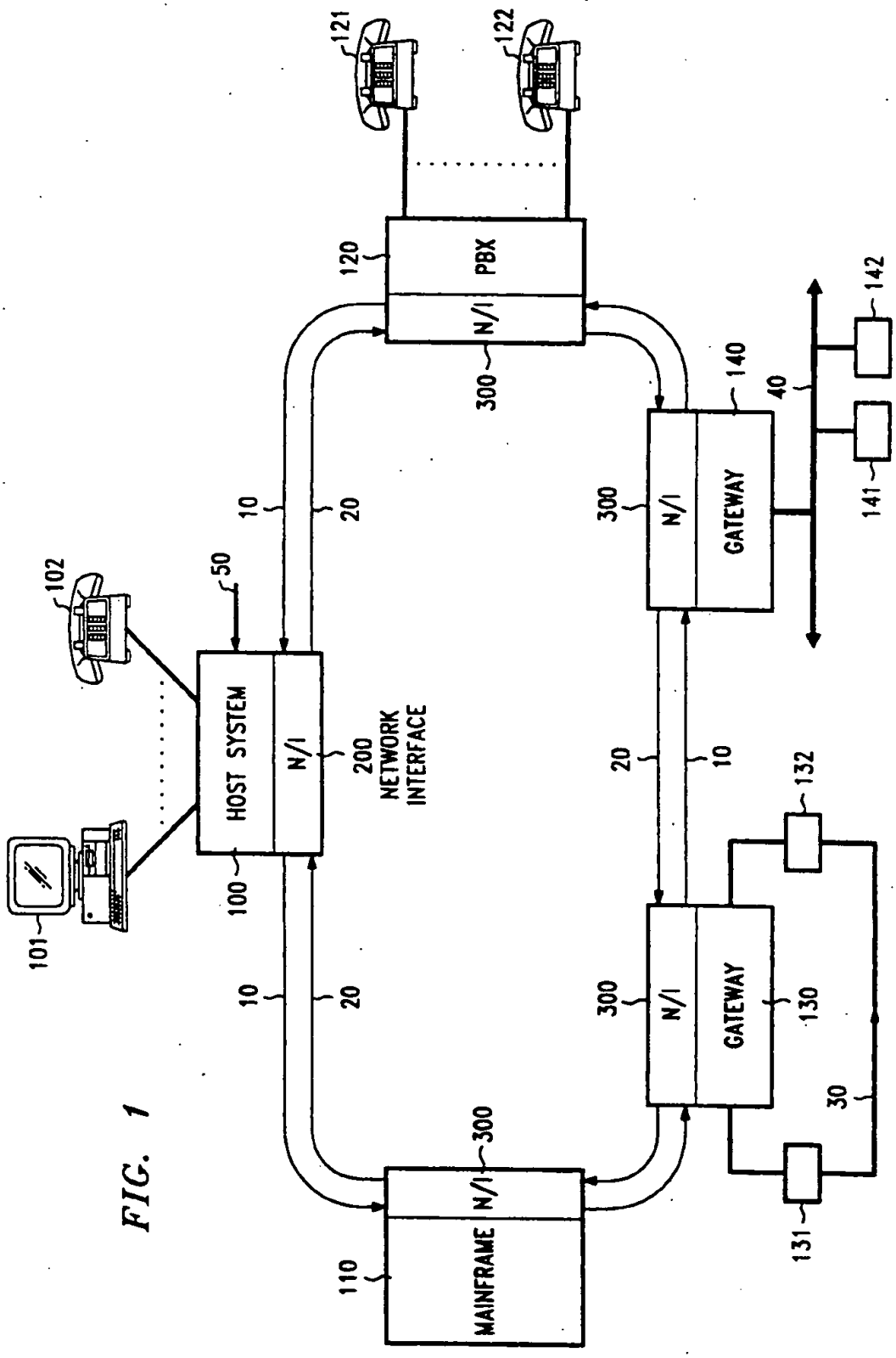

121
122
PBX
N/I
120
300
140
142
40
141
GATEWAY
N/I
300
10
20
50
102
HOST SYSTEM
N/I
200
NETWORK
INTERFACE
100
101
132
130
131
30
MAINFRAME
110

FIG. 1

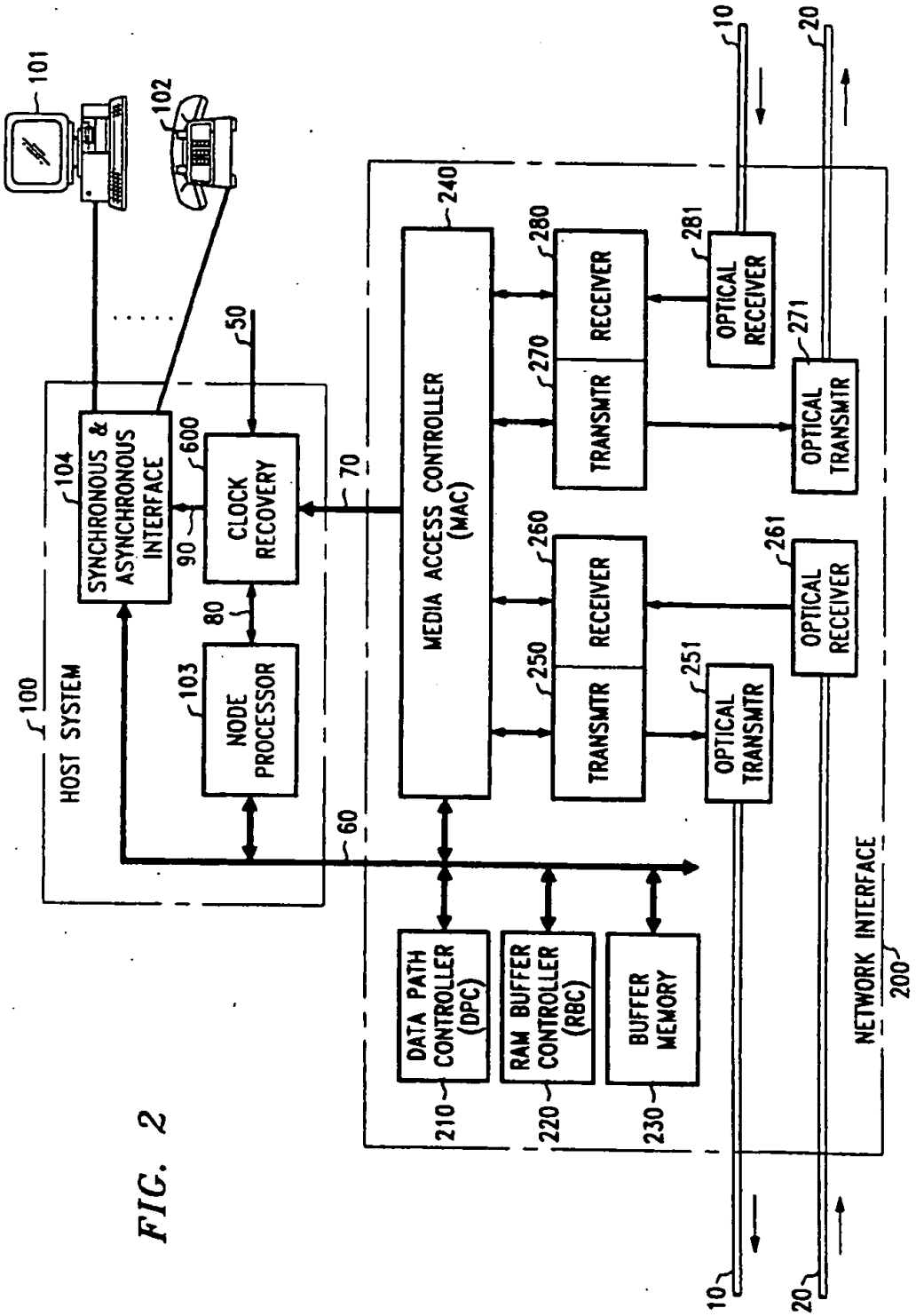

101
102
10
20
240
280
RECEIVER
281
OPTICAL RECEIVER
271
OPTICAL TRANSMTR
270
TRANSMTR
50
104
SYNCHRONOUS & ASYNCHRONOUS INTERFACE
600
CLOCK RECOVERY
70
MEDIA ACCESS CONTROLLER (MAC)
90
80
260
RECEIVER
261
OPTICAL RECEIVER
100
HOST SYSTEM
103
NODE PROCESSOR
250
TRANSMTR
251
OPTICAL TRANSMTR
60
DATA PATH CONTROLLER (DPC)
RAM BUFFER CONTROLLER (RBC)
BUFFER MEMORY
210
220
230
NETWORK INTERFACE
200
10
20

FIG. 2

FIG. 3

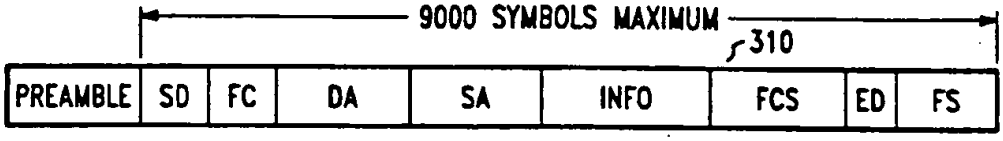


FIG. 4

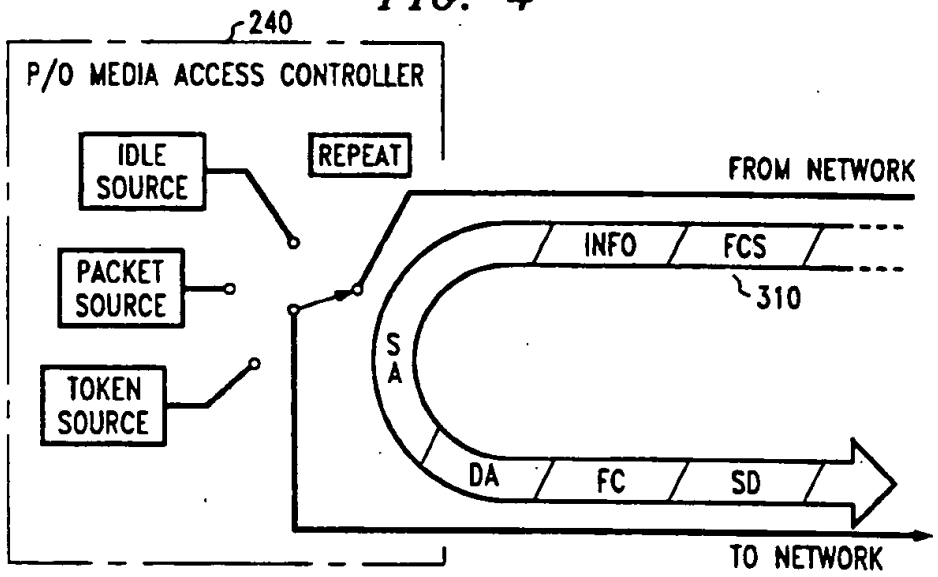


FIG. 5

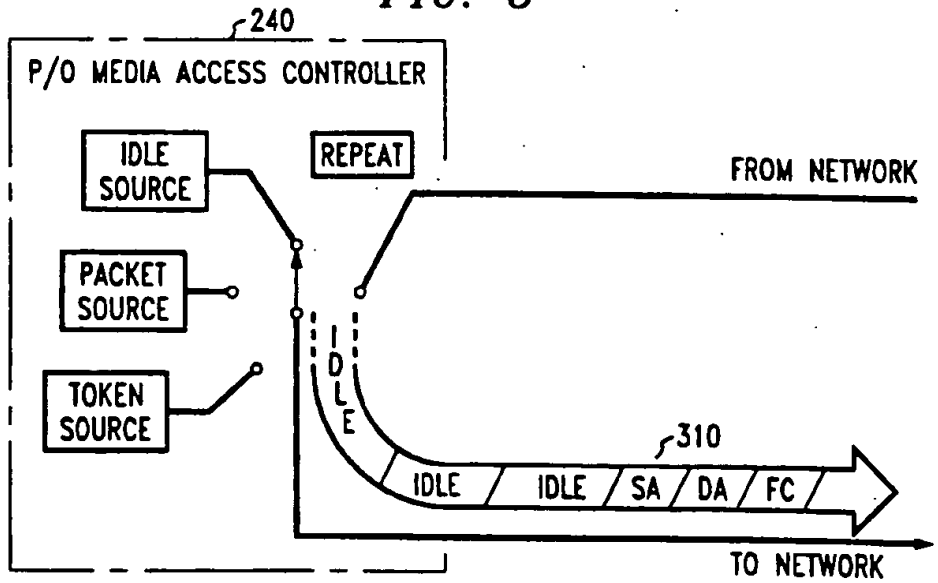
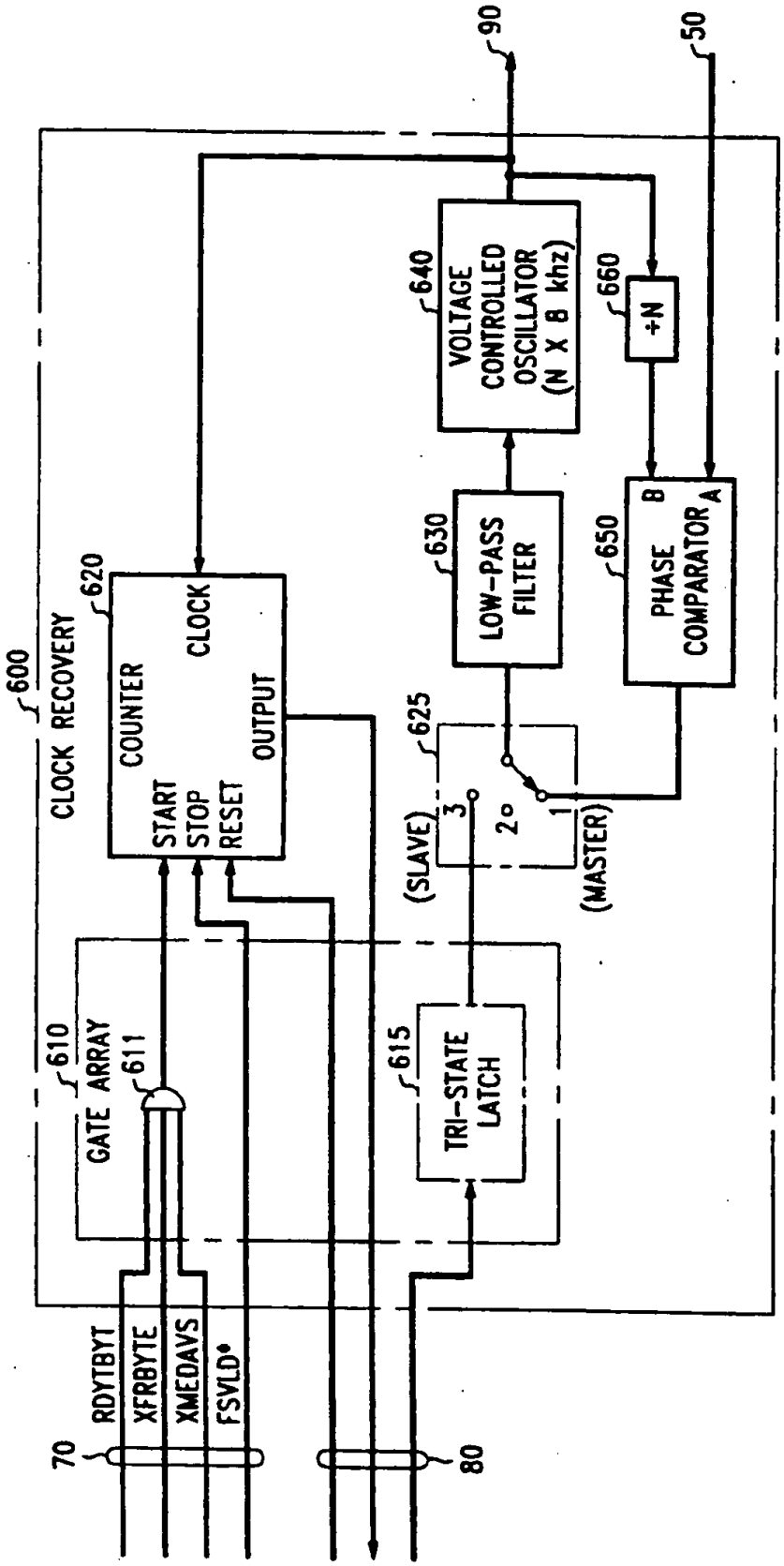


FIG. 6


90
50
640
VOLTAGE CONTROLLED OSCILLATOR (N X 8 khz)
660
+N
630
LOW-PASS FILTER
650
PHASE COMPARATOR
B
A
600
CLOCK RECOVERY
620
COUNTER
CLOCK
START
STOP
RESET
OUTPUT
625
(SLAVE)
3
2
1
(MASTER)
610
GATE ARRAY
611
615
TRI-STATE LATCH
RDYTBYT
XFRBYTE
XMEDAVS
FSVLD*
70
80

APPARATUS AND METHOD FOR TIMING DISTRIBUTION OVER AN ASYNCHRONOUS RING

TECHNICAL FIELD

This invention relates to a communication network having a plurality of nodes, and more particularly to a technique for distributing timing synchronization among selected nodes.

BACKGROUND OF THE INVENTION

Surprisingly, as computers grow in both power and proliferation, so too does their need to borrow and share more data with other computers. This need to exchange greater amounts of information can no longer be fully satisfied by the periodic data transfer between two computers but, rather, requires the simultaneous interconnection among a number of them, each having a particular specialization yet drawing from the specialization of the others. These interconnections are known as networks, and while they are limited in size and found in only corporate environments today, vast global geodesic networks will connect millions of islands of information tomorrow.

Communication systems which allow data transfer over telephone lines at a few hundred bits per second have been an integral part of computer systems for the past few decades. Only recently have networks capable of handling several million bits per second been widely available. Local area networks (LAN) have typically offered between 100 Kb/s and 10 Mb/s among a few hundred stations, and have been limited to a local area (a kilometer or so). One such network, Ethernet, is synchronous and operates at 10 Mb/s. Because rapid information transfer is indispensable in our highly competitive society, Ethernet is being superseded by a higher capacity network knwon as the Fiber Distributed Data Interface (FDDI) which transmits 100 Mb/s of data over each of two counter-rotating rings. FDDI can tolerate a separation of up to 2 kilometers between stations, and support and a total cable distance of 100 kilometers around a ring with 500 station attachments. FDDI possesses enough bandwidth to support up to 800 voice channels or perhaps 1-2 digitized video channels. One problem with voice or video traffic over FDDI, however, is that the network and interface are asynchronous, thereby preventing timing information from passing across the network boundaries. Although FDDI uses a Timed Token Protocol to provide both synchronous and asynchronous service, no technique has emerged as being clearly superior for clock synchronization at various stations around the ring. Whereas packet switching is possible over an asynchronous network, circuit switching requires a synchronous network and for that reason a synchronous FDDI network (FDDI-II) has been proposed. Unfortunately, FDDI and FDDI-II are incompatible, which is to say that a node adapted for FDDI-II operation cannot be part of an FDDI ring.

In an asynchronous system, each link requires its own clock. This means that each link is frequency- and phase-asynchronous vis-a-vis all other links in the ring, and that timing information cannot be recovered from the bit stream. Accordingly, the distribution of synchronous information, such as conventional telephone conversations, on the FDDI system, has certain inherent problems. One well-known solution for transmitting

synchronous information over an asynchronous facility is the use of elastic storage registers to buffer the differences in the bit rate. That is, data are written into a shift register at a first bit rate and read out of the shift register at a second bit rate. When packet information is being transmitted, it makes little difference whether the read and write rates are slightly different because packets are generally limited in size and the elastic storage registers can be made as large as desired. However, when transmitting continuous synchronous data, the elastic storage registers will overflow or underflow with the undesirable result that transmitted information will either be lost (overflow condition), or that incorrect information will be created (underflow condition).

U.S. Pat. No. 4,866,704 was issued on Sep. 12, 1989 and discloses a fiber-optic voice/data network. This patent teaches a technique for synchronizing a local clock by monitoring the average fill of an elastic storage register (receiving buffer), speeding up the local clock when the average fill is increasing, and slowing the clock when the average fill is decreasing so that overflow and underflow are prevented. While this technique is useful, it requires that synchronous data be continuously present to maintain synchronization.

It is also known to distribute a reference timing signal over a separate link to each node in a network, including a ring network. However, such a technique requires the installation of a separate network just for timing—thus defeating the structural simplicity of the ring and adding to its cost. It is therefore desirable to improve upon the prior art systems for distributing timing information over an asynchronous ring.

SUMMARY OF THE INVENTION

In accordance with the invention, a ring network includes a master node and one or more slave nodes where data enter and exit the network. Each of these nodes includes a clock for supplying timing information to equipment at that node. The master node measures the time delay encountered by data traversing the network, and transmits this measurement to the slave node. At the slave node, a similar measurement of time delay is made and compared with the measurement transmitted by the master node. Differences between these two measurements are used to adjust the frequency of the clock at the slave node.

In the illustrative embodiment of the invention, clock synchronization is optionally and independently available to any node in the ring network. As a result, the ring network simultaneously supports both synchronous and asynchronous data traffic. Accordingly, the primary reason for developing a synchronous FDDI network (FDDI-II) no longer obtains.

BRIEF DESCRIPTION OF THE DRAWING

The features and capabilities of the present invention will be more fully understood when reference is made to the detailed description and the drawing, of which

FIG. 1 illustrates a ring network having counter-rotating rings and a plurality of nodes, each serving different host system equipment;

FIG. 2 discloses equipment used at a particular node in accordance with the invention;

FIG. 3 shows a data packet such as used in the Fiber Distributed Data Interface (FDDI);

FIG. 4 illustrates the operation of a Media Access Controller at a node when repeating a data packet to the network;

FIG. 5 illustrates the operation of the Media Access Controller at a node when supplying an idle data packet to the network; and

FIG. 6 discloses, in block form, the details of clock recovery in accordance with the invention.

DETAILED DESCRIPTION

An emerging standard (ANSI X3T9.5)—better known as the Fiber Distributed Data Interface (FDDI)—defines a 100 Mb/s, time-token protocol. This protocol is to be implemented on an asynchronous, fiber-optic network optimized for 1300 nanometer technology. Fiber-optic communication is best suited for point-to-point transmission and is present in two local area network (LAN) topologies, the active hub star, and the ring. Active stars require extensive connections to wherever the central hub is located and introduce a single failure point that can disable the entire LAN. Single-ring networks are like chains which are vulnerable to failures at any station. FDDI minimizes this by using dual, counter-rotating rings. This provides an alternate path if a station or a link fails, allowing data to be wrapped back onto the secondary ring in the event of failure. The secondary ring may be only a standby ring, or it may be used for concurrent transmission thereby providing a 200 Mb/s network. Additional background information regarding FDDI is contained in an article by Floyd E. Ross entitled: *FDDI—a Tutorial*, published in the IEEE Communications Magazine, May 1986–Vol. No. 24, No. 5 at pp. 10–17.

Referring now to FIG. 1, there is disclosed an example of an FDDI network comprising primary ring **10** and counter-rotating secondary ring **20**. Nodes are illustratively positioned at locations around the network that may be separated by large distances. Associated with each node is a host system that is available to users at all other nodes via the fiber-optic ring. Together they form a network whose combined resources are formidable and diverse. A generalized host system **100** is shown connecting stations **101, 102** to the FDDI network via network interface (N/I) **200**. In the discussion that follows, network interface **200** and host system **100** are designated as the "master node". In the present invention, the master node distributes timing information to all of the other nodes. Accordingly, network interface **200** is similar to the other network interfaces **300** in all respects except that it supplies timing information to them. Network interface **200** receives a timing signal over line **50** via host system **100**. Any pair of nodes desiring to exchange synchronous data with each other need to synchronize their clocks. This can be accomplished by adjusting the clock frequency at one of the nodes to agree with the other, or by adjusting the clock frequency of both of the nodes to agree with a reference clock. Once synchronization is achieved, information such as digitized voice can be transmitted and received by appropriate equipment at these nodes.

Host system **100** can be any mainframe, workstation, minicomputer or peripheral to which a network interface is attached. Its role is to provide and receive network data and perform some higher-level protocol functions which are not handled by the associated stations. Examples of other host systems include, but are not limited to, mainframe **110**, PBX **120**, and gateways, **130, 140**.

Stations **131, 132** are shown connected to gateway **130** by, for example, a network **30** (e.g., IEEE-802.5 token ring). Stations **141, 142** are shown connected to gateway **140** by, for example, a bus **40** (e.g., IEEE-802.3 Ethernet or IEEE-802.4 token bus). Stations **121, 122** are shown connected to PBX **120**. Each of these nodes connects to the fiber-optic ring by a network interface (N/I) **300** whose operation is defined in detail by FDDI standards, but will be briefly discussed hereinafter.

FIG. 2 discloses the equipment needed at each node to support interconnection with the FDDI network. Optical fibers **10, 20** are made from glass-clad silica and consist of an inner core surrounded by a glass cladding (having a different refractive index than the silica) and a protective sleeve. Plastic cladding is not used in FDDI applications because of its greater attenuation. Optical receivers **261, 281** receive modulated light-wave signals, whose center wavelength lies between 1270 and 1380 nanometers, from optical fibers **20, 10** respectively, and consist of a photodetector, an amplifier, a shaping filter, a comparator, a buffer to provide suitable electrical levels, and a signal detect circuit. The photodetector is constructed from an InGaAs/InP PIN photodiode that senses light intensity and converts it into current pulses. Optical transmitters **251, 271** transmit modulated light-wave signals, whose wavelengths are also in the 1300 nanometer range, to optical fibers **10, 20** respectively. Each optical transmitter consists of a driver and a light emitting diode (LED). The LED is typically made of InGaAsP, and generates light whose intensity is a function of the input data to be transmitted.

Transmitter/receiver pair **250, 260** controls the encoding and decoding of data and control symbols, serializing the data clock recovery, line-state detection and reporting. In operation, transmitter **250** serializes eight-bit-wide parallel data from Media Access Controller (MAC) **240** to produce non-return-to-zero (NRZ) code. 4B/5B coding is used which comprises translation of 4-bit groups of data into a 5-bit value, and conversion into the NRZ format for transmission on the fiber-optic ring. 4B/5B encoding is said to be 80% efficient since a 100 Mb/s data rate translates into a 125 megabaud rate which is present on the fiber-optic ring. Receiver **260** decodes the data received from the fiber-optic ring and converts it into symbols that can be recognized by the MAC. Tasks of receiver **260** include retiming the data to an internal clock via a phase-locked loop and an elastic buffer, and converting 5-bit code back into 4-bit code. Differential Manchester coding is frequency compared with 4B/5B coding and has the advantage of being rich in clock information with a transition at every bit, but the disadvantage of being only 50% efficient. Suitable devices for the transmitter and receiver are the AM7985 and AM7984, respectively, which are commercially available from Advanced Micro Devices, Inc. Transmitter **270** and receiver **280** are identical to transmitter **250** and receiver **260**, but are dedicated to different optical fibers.

Bus **60** comprises several buses. It includes, for example, a 32-bit bus used to interconnect buffer memory **230** with other devices, while several 16-bit buses are used between controllers **210, 220** and **240** and node processor **103**.

Data Path Controller (DPC) **210** converts data in received packets from byte-wide to 32-bit parallel word formats, performs parity checks and generates packet and node status. A suitable device for the DPC is the AM79C82.

RAM Buffer Controller (RBC) 220 generates addresses to buffer memory 230 for received and transmitted packets. The RBC handles buffer management and arbitrates direct memory access coming from DPC 210, node processor 103 and other host system apparatus. A suitable device for the RBC is the AM79C81.

Buffer memory 230 is a conventional 256-kbyte static random access memory which is readily available from a number of commercial sources. The buffer memory is a storage area accessible by the controllers 210, 220, 240, the node processor 103 and other host system equipment.

Media Access Controller (MAC) 240 controls the right to transmit data to the network based on capturing a token according to a set of network rules, recognizes addresses, controls ring recovery, and handles network and frame status. A suitable device for the MAC is the AM79C83. The above devices designated 210, 220, 240, 250, 260 are all commercially available from Advanced Micro Devices, Inc. in what is known as the SUPERNET TM chip set which has been designed to conform to the FDDI standard.

Node processor 103 is a separate microprogrammed or conventional microprocessor-based system used for offloading the host and overseeing the operation of the network interface 200. The node processor communicates with the network interface over bus 60. A suitable device is the AM29000 RISC Processor, also commercially available from Advanced Micro Devices, Inc., which includes a number of embedded counters. One of these counters is used as counter 660 in FIG. 6, and is shown functionally separated from the node processor for greater clarity in explaining the invention.

Block 104 (Synchronous and Asynchronous Interface) contains the circuitry needed to buffer information between memory 230 and a synchronous Time Division Multiplexed (TDM) bus. Here, terminal 101 and telephone 102 represent the kinds of devices that might be connected to such a bus. Block 104 uses Direct Memory Accessing to move data between memory 230 and another buffer memory within block 104. In addition, block 104 contains the circuitry needed to move data to and from bus 60 and the TDM bus. A source of synchronous timing is provided to block 104 over line 90. Since the present invention is concerned with establishing timing, such as present on line 90, rather than its use, further details regarding block 104 are omitted for clarity.

FDDI Protocol

Information is transmitted on the FDDI ring as packets of data, each having a maximum size of 9000 symbols (4 bits/symbol) and only one node may transmit a data packet at a time. However, in order to avoid anarchy, a single token is passed from node to node giving the token holder an exclusive right to transmit data. FDDI protocol promotes fair an deterministic access to network resources for all stations. This is done using a timer that measures the time between token arrivals, a timer that controls how long a token can be held for transmission, and a counter that indicates the number of times the token arrives later than expected. The rules for data transmission on FDDI depend upon the type of data to be sent. For transmission of synchronous data, the rules are straightforward. The amount of synchronous data that can be transmitted at a given token opportunity is limited by the bandwidth-allocation process which assures that if every station transmits its maximum allotment of synchronous data, the negotiated target token-rotation time (TTRT) will not be exceeded. The transmission of asynchronous data is slightly more complicated, since the maximum time allotted for transmission is not constant. Instead, a station can transmit asynchronous data until the unused bandwidth on the ring is exhausted.

MAC 240 controls the flow of data transmission on the ring, and indicates when a data packet is transmitted as well as when it has returned from a trip around the network. This feature is highly important in the practice of the invention as will be pointed out later. FIG. 3 discloses the format of an FDDI packet 310. Packets are preceded by a PREAMBLE having a minimum of 16 "idle" control symbols. Packets begin with a Start Delimiter (SD) composed from the J and K control symbols of the 4B/5B code. This is followed by a 2-data-symbol Frame Control (FC) field that identifies the type of packet. The Destination Address (DA) identifies the intended recipient of the packet. Likewise, the Source Address (SA) identifies the packet's sender. Addresses can be either 26 or 48 bits in length. The DA field can point to a single station, a group of stations, or all stations on the ring. Following SA comes the variable length information field (INFO). A frame Check Sequence (FCS) field contains 4 bytes of data that are the result of a 32-bit Autodin II cyclic redundancy check polynomial. The FCS ensures the data integrity of the FC, DA, SA, INFO, and FCS fields. Following the FCS field, an End Delimiter (ED) formed with the "T" symbol is transmitted. The Frame status (FS) field is used for symbols that determine whether the packet was received with error, if the address was recognized, or whether the packet was copied. The maximum length of the packet (here 9000 symbols) is limited by the length of the elastic buffer and the worst-case frequency difference between the clocks at two nodes

Referring now to FIG. 4, it will be seen that MAC 240 acts like a switch. Normally the switch selects a source of IDLE control symbols for transmission on the ring. When the Start Delimiter arrives, MAC 240 switches to a repeat path. The packet is monitored, copied if required, and simultaneously repeated. MAC 240 can also switch to source its own packets or issue a token. Packets are removed from the ring by the originating station. This process, called "stripping", is illustrated in FIG. 5. The MAC repeats the packet until the SA field is received. Upon recognition of the Source Address, the switch moves to the IDLE position. The resulting packet fragment on the ring is ignored and eventually removed when it reaches a station holding the token for transmission. A station wishing to transmit must first "capture" a token. The token is a 6-symbol packet that is uniquely recognized. A station captures the token by performing the stripping action. Only the token SD field is repeated on the ring. Once the token is captured, the station can begin transmitting packets. When the last packet is sent, the station immediately follows by issuing a new token.

It is noted that FDDI is asynchronous, which means that the clock at each of the various nodes is not synchronized with a common oscillator. Such synchronization frequently comes at a very high price. It is important, however, that each of the nodes be equipped to receive and transmit data at an agreed-upon rate, and that data formats be established for efficient communication. FDDI employs an interesting solution to the ring clocking problem. The total ring, including all of

its stations and all of its links, must continue to have the same apparent bit length during data transmission. Otherwise, some bits would be lost or gained as a frame was repeated around the ring. In the face of jitter, voltage, temperature, and aging effects, such stability can only be realized through special provisions. At each node, an elastic buffer is inserted between the receiver, which employs a variable frequency clock to track the clock of the previous transmitting station, and a transmitter, which runs on a fixed-frequency clock. The elastic buffer at each station, is re-initialized during a preamble which precedes each frame or token. This has the effect of increasing, or decreasing, the length of the preamble which is initially transmitted as 16 or more symbols. The transmitter clocks typically have 0.005% stability and the elastic buffer stores 10 bits of data. Accordingly, while significant data can be transmitted without exceeding the limits of the elastic buffer, practical tradeoffs are made between the accuracy of the clock and the length of the buffer.

FIG. 6 discloses detail regarding the construction of clock recovery circuit 600 which is used at both master and slave nodes to provide a source of timing on line 90 to all equipment at the node which it serves. Switch 625 is used for selecting between operation as a master timing source (position 1), a free-running timing source (position 2), or a slave timing source (position 3). Because digital voice communication equipment at telephone offices typically use 8 kHz as the sampling frequency when carrying out analog-to-digital conversion, 8 kHz is the information rate for synchronous communications. Accordingly, voltage-controlled crystal oscillator (VCO) 640 operates at a multiple (N) of the 8 kHz base frequency so that it can be easily synchronized therewith. In an illustrative embodiment, VCO 640 operates at 8.192 MHz, and N=1024. Synchronization is not required at any node which operates asynchronously, although synchronization requires very little in the way of additional components. For example, in the preferred embodiment of the invention counter 620 is actually part of node processor 103 (see FIG. 2), and gate array 610 only adds a few gates to a larger gate array (not shown) used for performing a number of other tasks within the host system. In the Free-Run Mode, (position 2 of switch 625) VCO 640 provides an 8.192 MHz crystal-controlled timing signal to all using equipment. VCOs are well-known devices that are available from a number of vendors including CTS Corporation and SaRonix.

Synchronizing VCO 640 to an 8 kHz signal, such as provided by a telephone office, is accomplished when switch 625 is in position 1 where operation as the master source of timing for the ring network is desired. Divide-by-N counter 660 functions to divide the 8.192 MHz signal by 1024 to provide an 8 kHz square wave whose phase is compared with the phase of a reference 8 kHz signal on input line 50. Phase comparator 650 is a conventional phase-locked-loop, such as the 74HC4046M, which is a commercially available device from a number of vendors including Signetics and National Semiconductor Corporation. The output of the phase comparator is a measure of the error in the frequency of the VCO. This error signal comprises a binary signal whose logic levels are used to increase/decrease the frequency of VCO 640. Alternatively, a phase comparator having output states other binary voltages may be advantageously used in the present invention. Indeed, the following paragraph discloses a Tri-State Latch 615 that provides an output state that neither increases nor decreases the frequency of the VCO. When the "A" and "B" inputs to the phase comparator are identical, the VCO is at the desired frequency, although the output signal alternates between the two logic levels. Low-Pass filter 630 is used to smooth such time-varying signals at the input to VCO 640 so that an averaged voltage is presented to the VCO.

Synchronizing VCO 640 to timing information supplied by a remote source of timing is accomplished when switch 625 is in position 3 where operation as a slave node is desired. The fundamental principle adapted for use in the present invention is simply stated: independent observes measuring the identical phenomenon should reach an identical result. If not, then the measuring apparatus is modified until the results are identical. In the present invention, the phenomenon to be measured is the time delay that a data signal encounters during a single trip around the ring network. Since the ring network is identical for all observers, then they should all measure identical time delays—but they do not. And the reason that they do not is because the clocks that are used to make the measurements are different. The present invention designates one observer (master node) to make the "official" measurement of time delay around the ring (ring latency), and to communicate this measurement to all nodes. Any node seeking to become synchronous with the master node makes its own measurement of ring latency, compares it with the "official" measurement made by the master node, and adjusts its clock so that its measurement of ring latency is identical with that of the master.

Returning now to FIG. 6, adjustments to VCO 640 are made based on differences in ring latency measurements. At this time, switch 625 is in position 3 (Slave Mode). AND gate 611 responds to three input signals from line 70 to provide a start pulse for counter 620. When a synchronous data packet leaves Media Access Controller 240 (see FIG. 2), all three inputs to AND gate 611 are high causing its output to also become high. Using the illustrative AM79C83 device, these leads are designated RDYTBYT, XFRBYTE and XMEDAVS. It is noted that this synchronous packet is generated by the node, and not a repeated synchronous packet!

Another signal lines FSVLD*, also from MAC 240, signals when the node has received its own packet—which is to say that the transmitted packet has traversed the entire ring and needs to be stripped from the ring. This signal (FSVLD*) is used to stop counter 620 so that the number which is present at the output of the counter is the number of cycles of the 8.192 MHz clock that have occurred between the start and stop pulses and is a measure of ring latency delivered to node processor 103 (see FIG. 2) which appends this number to the end of the next synchronous packet that is to be sent around the ring. This is done regardless of whether the node is the master or a slave. If the node is the master, the process is ended. If the node is a slave, the node processor compares the value that is read from counter 620 to the value that the master node transmitted on its last synchronous packet. These values are subtracted from each other in the node processor and the results of the subtraction are delivered to latch 615. In the preferred embodiment of the invention, latch 615 provides three output states comprising +5 volts, ground, or an open circuit. These states are used to either increase, decrease, or not affect the frequency of the voltage-controlled crystal oscillator 640.

Although a particular embodiment has been shown and described, it is understood that various modifications are possible within the spirit and scope of the invention. These modifications are possible within the spirit and scope of the invention. These modifications include, but are not limited to, use of the invention on non fiber-optic networks, ring latency measurements in which the data packet makes more than a single excursion around the ring, use of the invention in a network where packets, per-se, are not used, and use of the invention in configurations where node connections are parallel rather than serial. Furthermore, it is understood that in a network of counter-rotating rings, either ring may be used for ring latency measurements and either ring may be used to transmit the results of such measurements. While it is preferred that the master and the slave nodes each use the same ring to make latency measurements, it is not required.

I claim:

1. A communication network interconnecting a master node and at least one slave node in a ring topology, each node including a clock for providing timing signals to equipment at that node, each node further including means for transmitting and receiving data to and from the ring network,

characterized in that:

the master node includes:

means responsive to the clock at the master node for measuring a first time delay encountered by data traversing the ring network;

means for transmitting the measure of the first time delay to the slave node,

the slave node includes:

means responsive to the clock at the slave node for measuring a second time delay encountered by data traversing the ring network;

means for receiving the measure of the first time delay from the master node; and

means responsive to the difference between the measured first and second time delays for adjusting the frequency of the clock at the slave node to bring its frequency into a predetermined relationship with the frequency of the clock at the master node.

2. The invention of claim 1 wherein the measure of the first time delay is transmitted to the slave node over the ring network.

3. The invention of claim 1 wherein time is measured by counting the number of clock cycles that occur between the time when a data packet enters the ring network and the time when that packet exits the network, said data packet being transmitted and received by node performing the measurement.

4. The invention of claim 1 wherein the predetermined relationship between the frequency of the clock at the slave node and the frequency of the clock at the master node is equality.

5. The invention of claim 1 wherein the communication network comprises fiber-optic links between the nodes.

6. The invention of claim 5 wherein the communication network conforms to the Fiber Distributed Data Interface protocol.

7. A local area network, configured as a token-passing ring, for communicating packets of information between nodes attached to the ring, at least one of the nodes being designated slave for the purpose of receiving information regarding the frequency of the clock at the master node, the slave node

characterized by:

means for transmitting an identifiable packet onto the ring;

means for receiving said identifiable packet after it has traversed the ring;

means responsive to a clock at the slave node for measuring the time interval between the event that the identifiable packet is transmitted and the event that it is received in order to measure ring latency;

means for receiving a measurement of ring latency made at the master node;

means jointly responsive to the ring latency measurement made at the master node and to the ring latency measurement made at the slave node for modifying the frequency of the clock at the slave node in a manner that decreases the difference between the latency measurements.

8. A method for timing synchronization in an asynchronous ring network that interconnects first and second stations; each station having a timing signal generator, a transmitter for entering data onto the network, and a receiver for receiving data from the network, the ring network circulating data around the ring in a time interval known as "ring latency", the method comprising the steps of:

measuring ring latency at the first station;

transmitting the measure if ring latency to the second station;

measuring ring latency at the second station;

calculating an error signal, at the second station, as the difference between measurements of ring latency made by said first and second stations; and

modifying the frequency of the timing signal generator, at the second station, in a manner that decreases the magnitude of the error signal.

9. The method of claim 8 wherein the step of measuring ring latency at a station comprises the following steps:

transmitting a data packet onto the ring network;

identifying the return of the data packet from the ring network; and

measuring the time interval between steps of transmitting and receiving said data packet.

10. The method of claim 9 wherein the station includes a counter that counts pulses from the timing signal generator, and wherein the time interval measurement comprises the following steps:

resetting the counter prior to the time interval measurement;

starting the counter when the data packet is transmitted; and

stopping the counter when the data packet is received.

11. A method for establishing frequency synchronization between clocks at two different geographic locations, each location having access to a common transmission media, and each location having apparatus, including its clock, for independently measuring signal propagation delay over the transmission media, the method including the steps of:

measuring the propagation delay over said transmission media at each location:

communicating the measurement of the propagation delay made at one location to the other location; and

changing the frequency of the clock at said other location in accordance with the difference between

the measurements of propagation delay made at each location.

12. The method of claim 11 wherein the common transmission media comprises communication links interconnecting a plurality of nodes in a ring network, and wherein said two different locations correspond to two of the nodes.

13. The method of claim 11 wherein the common transmission media comprises fiber-optic cables interconnecting said locations in a ring topology.

* * * * *

US005878044A

United States Patent [19]

Frischknecht et al.

[11] Patent Number: 5,878,044

[45] Date of Patent: Mar. 2, 1999

[54] DATA TRANSFER METHOD AND APPARATUS

[75] Inventors: Deborah Ann Frischknecht, Kanata; John Frank Pillar, Nepean; Alan Charles Coady, Ottawa; Jonathan David Loewen, Kanata, all of Canada

[73] Assignee: Northern Telecom Limited, Montreal, Canada

[21] Appl. No.: 637,961

[22] Filed: Apr. 25, 1996

[51] Int. Cl.[6] H04J 3/16

[52] U.S. Cl. 370/466; 370/469

[58] Field of Search 369/298, 401, 369/402, 498, 502, 503, 535, 911, 537, 538, 465, 469, 466, 395, 464; 375/220, 257, 369, 370, 377

[56] References Cited

U.S. PATENT DOCUMENTS

4,071,887	1/1978	Daly et al.	375/377
4,736,394	4/1988	Giovanelli et al.	375/377
5,267,263	11/1993	Feezel et al.	375/220
5,619,499	4/1997	Nakabayashi	370/469

OTHER PUBLICATIONS

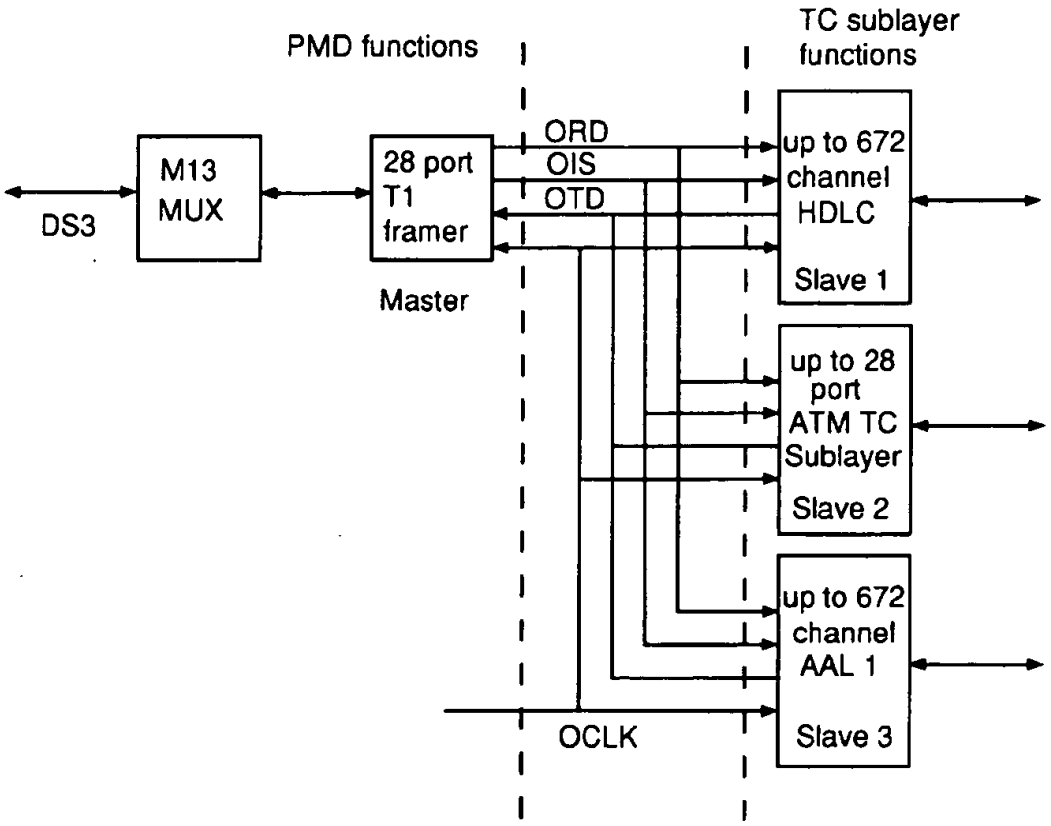
"Four Channel DS1 Framer", Eugene L. Parrella et al., IEEE International ASIC Conference and Exhibit, Mar. 1994, pp. 445–448.

Primary Examiner—Chau Nguyen

[57] ABSTRACT

The invention resides in the field of transferring data and other information from multiple asynchronous TDM channels across a synchronous interface in digital blocks of a preset length. Presently, transfer of data and timing information requires serial transmission of frame payload, qualified with clock and frame pulse indications. A separate set of these signals is required in both directions for each port or physical link. An interface of the invention permits data transfer in blocks which are uniquely identified for a specific port or links and the position of the block within the frame (if the transfer is one of framed data). The invention greatly reduces the number of required signals, thus enabling reduction of pin count requirements and an increase in the number of serviced ports.

13 Claims, 4 Drawing Sheets



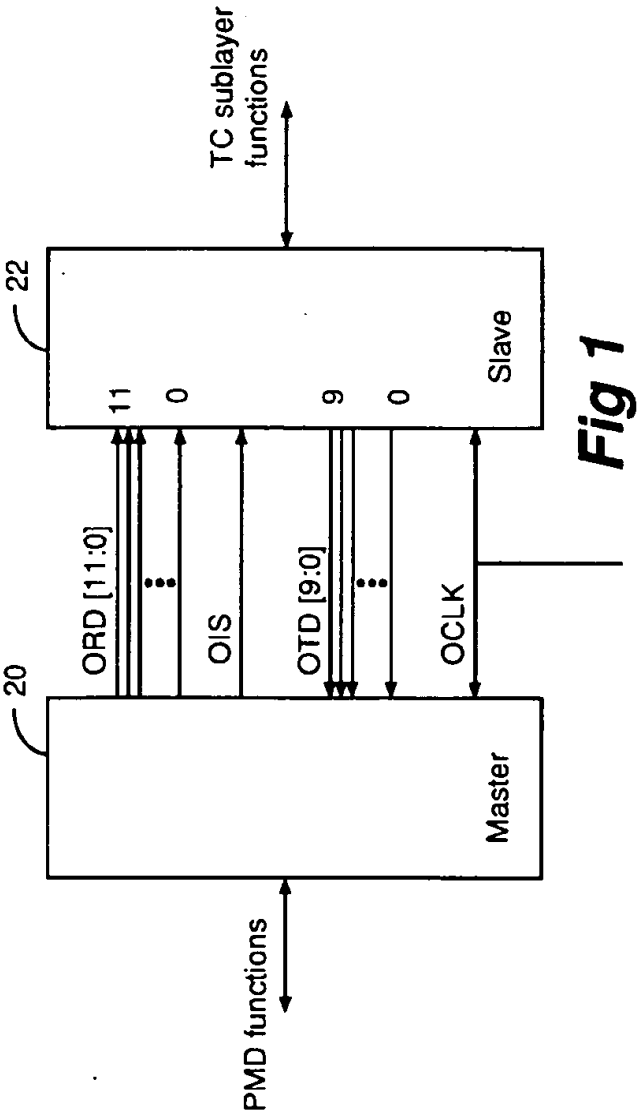

TC sublayer functions
22
Slave
11
0
9
0
ORD [11:0]
OIS
OTD [9:0]
OCLK
20
Master
PMD functions

Fig 1

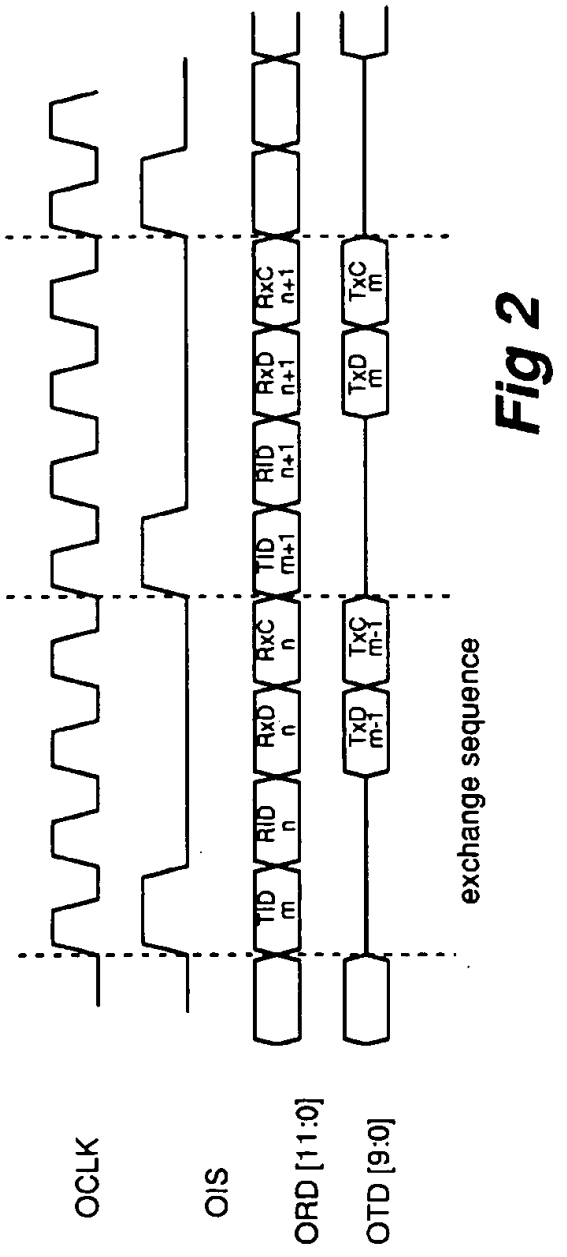

OCLK
OIS
ORD [11:0]
OTD [9:0]
TID m
RID n
RxD n
RxC n
TID m+1
RID n+1
RxD n+1
RxC n+1
TxD m-1
TxC m-1
TxD m
TxC m
exchange sequence

Fig 2

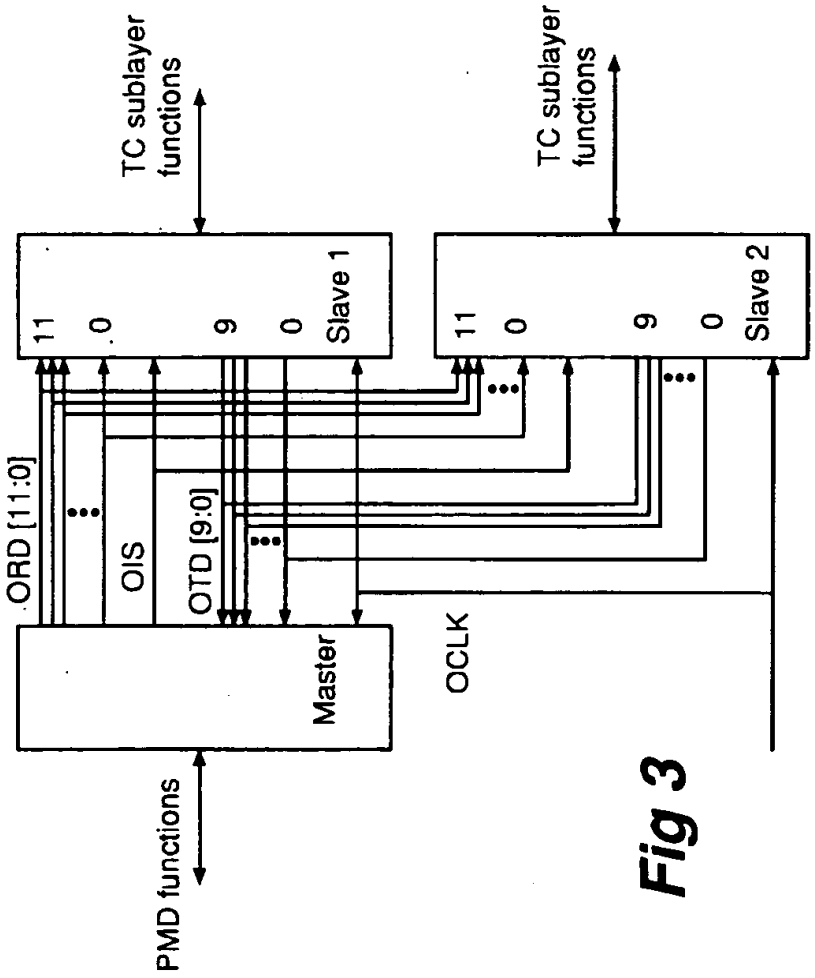

TC sublayer functions
TC sublayer functions
Slave 1
Slave 2
11
0
9
0
ORD [11:0]
OIS
OTD [9:0]
OCLK
Master
PMD functions

Fig 3

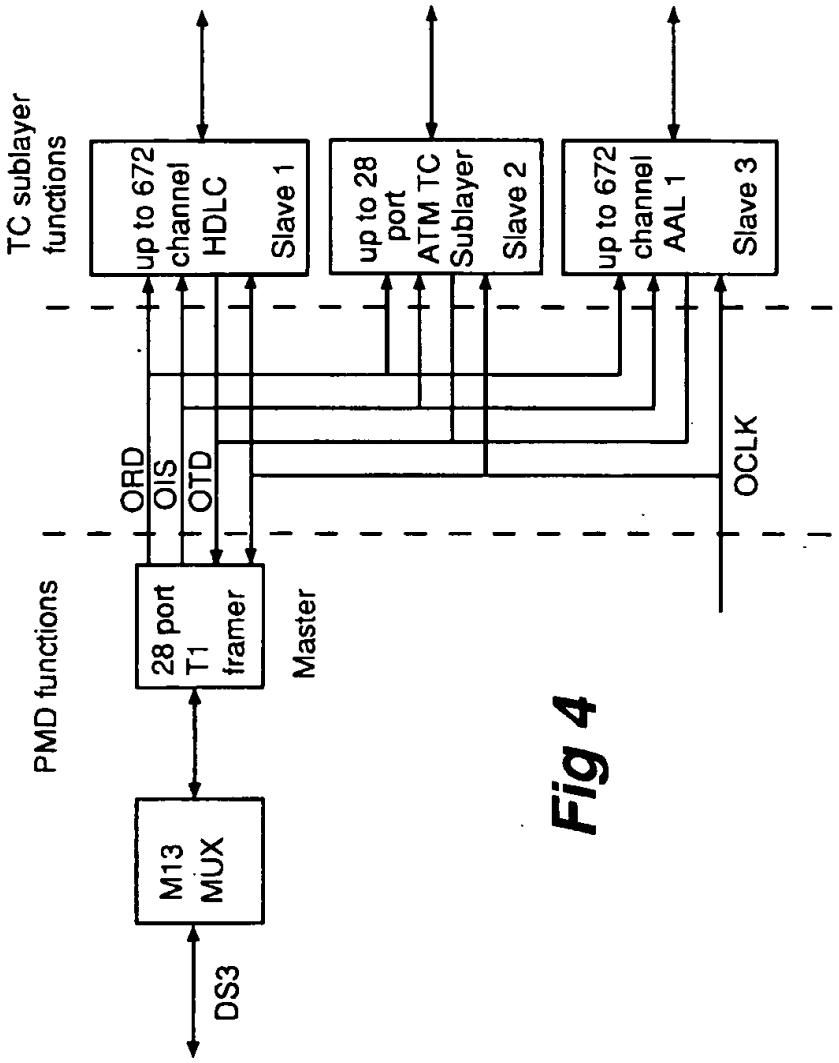

TC sublayer functions
up to 672 channel HDLC
Slave 1
up to 28 port ATM TC Sublayer
Slave 2
up to 672 channel AAL 1
Slave 3
ORD
OIS
OTD
OCLK
PMD functions
28 port T1 framer
Master
M13 MUX
DS3

Fig 4

DATA TRANSFER METHOD AND APPARATUS

FIELD OF THE INVENTION

The present invention relates generally to interfacing non synchronized digital signal streams which exist in different sublayers of the physical layer. In particular, it is directed to interfacing of digital signal streams by way of a master and a slave exchanging blocks of data stream, each comprising a predetermined number of bits.

BACKGROUND OF THE INVENTION

The current method of transfer of data and timing information between Physical Medium Dependent (PMD) and Transmission Convergence (TC) sublayers requires serial transmission of frame payload qualified with clock and frame pulse indications. A separate set of these signals is required in both directions for each port or physical link.

For example, digital signals are often multiplexed to produce higher bit rates for higher capacity transmission systems, as a means of utilizing the same transmission medium economically for many different users. This multiplexing is one of functions that perform such data transfer. Different digital signal hierarchies were developed in North America, Europe and other parts of the world. In North America, DS1,DS2 and DS3 whose bandwidths are respectively 1.544 Mbit/sec., 6.312 Mbit/sec. and 44.736 Mbit/sec. are used. Transmission services which provide these digital signals are called T1, T2 and T3. DS0 is a basic TDM signal (voice channel signal) of 64 Kbit/sec. Thus, 24 DS0 streams are multiplexed to one DS1,four of which are combined into one DS2. Seven DS2 streams are multiplexed to one DS3 stream. At each multiplexing operation, certain overhead bit or bits are added for framing, synchronization and other housekeeping functions. In some multiplexing operations, certain payload databits are borrowed for housekeeping functions as well.

According to "Four channel DS1 Framer" Eugene L. Parrella et al, IEEE International ASIC Conference and Exhibit:

> "Wideband telecommunications services such as T3, SONET, and inverse multiplexed T1 services are driving more highly integrated multichannel T1 cards. With the availability of single chip M13 (T1/T3) multiplexers and multichannel SONET mappers, T1 framers and T1 line interface units may become the bottlenecks to further reductions in card size, power and cost. A multichannel T1 framer offers substantial board area reduction, and is highly desirable if competitive in power consumption and cost to multiple single channel framers.
>
> As an example, a DS0-T3 switching application can be considered. A single chip M13 multiplexer is employed, multiplexing 28 DS1s into a DS3. To convert DS0's into 28 framed DS1's, 28 DS1 framers are required. A multichannel framer can be used effectively to lower parts count for the system."

Therefore, the above application of DS1–DS3 interface requires 112 connections between a group of 7 quad DS1 framers and a M13 multiplexer.

The present invention permits the transfer of data and timing information between the PMD and TC sublayers, a predetermined sized block of data at a time. Each data block is uniquely identified with an identification number which specifies the port or physical link with which it is associated and the position of the data block within the frame (if the transfer is one of framed data). Timing information is passed between the two sublayers in digital representation form. It represents an offset with respect to a reference clock which can be board, system, or network wide.

This invention greatly reduces the number of signals required between the two sublayer devices. This makes it physically possible to increase the number of ports serviced by PMD and TC sublayer devices by greatly reducing their pin count requirements.

In one embodiment, the present invention permits interfacing between digital signals of two different hierarchical levels very efficiently. In particular, as an example, in a DS1–DS3 interface only 24 lines (instead of 112 connections mentioned above) are required, running at a reasonable speed of 50 MHz.

Furthermore, for channelized applications, the necessity to demultiplex onto separate asynchronous bit streams only to multiplex back to one synchronous stream has been eliminated.

OBJECTS OF THE INVENTION

It is therefore an object of the invention to provide a method and an apparatus for transferring data, timing and other control information between two sublayer devices efficiently.

It is another object of the invention to provide a method and apparatus for interfacing two sublayer devices which require less signals for transferring data and other information.

SUMMARY OF THE INVENTION

Briefly stated, according to one aspect, the invention is directed to a method of synchronously transferring data and other information between a physical medium sublayer and a transmission convergence sublayer, each handling a non synchronized digital signal stream. The method comprises steps of a master bidirectionally handling a non-synchronized digital signal stream to/from the physical medium sublayer and a slave bidirectionally handling a non-synchronized digital signal stream to/from the transmission convergence sublayer. The method further includes steps of the slave and the master exchanging the data and other information in individually identifiable blocks of predetermined lengths during each successive exchange sequence of a preset length of time in a time synchronized fashion under a clock signal. The method comprises yet further steps of, during each exchange sequence, the master sending the slave identifiers of blocks of data and other information to be exchanged between the slave and the master, and in response to the identifiers, the slave exchanging with the master the blocks of data and other information so identified.

According to another aspect, the invention is directed to an interface for synchronously transferring data and other information between a physical medium sublayer and a transmission convergence sublayer, each sublayer handling a non-synchronized digital signal stream. The interface comprises a master for bidirectionally handling a non-synchronized digital signal stream to/from the physical medium sublayer and a slave for bidirectionally handling a non-synchronized digital signal stream to/from a transmission convergence sublayer. The interface further includes time division multiplexed buses connecting the master and the slave for exchanging the data and other information in individually identifiable blocks of predetermined lengths during each successive exchange sequence of a preset length

of time, in a time synchronized fashion under a clock signal, in that the identifiable blocks in each successive exchange sequence contain data and their identifiers.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 is a schematic illustration of an interface according to one embodiment of the invention;

FIG. 2 is a timing diagram of signals of one embodiment of the invention;

FIG. 3 is a schematic illustration of an interface according to another embodiment of the invention in which more than one slave is used; and

FIG. 4 is a schematic illustration of an interface according to yet another embodiment of the invention in which HDLC, ATM TC and AAL1 sublayers are shown as examples.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS OF THE INVENTION

Generally speaking, the invention concerns a synchronous interface used to transfer multiple bidirectional non-synchronized data streams between a PMD and the Physical Layer.

One embodiment of the invention is concerned with multiplexing digital signals of different hierarchy levels, in which 28 DS1 data streams are interfaced into one DS3 data stream. According to DS1 and DS3, the payload data are formatted in blocks of an integer multiple of octets. Conveniently, therefore, this can be called an Octet Stream Interface. This interface defines the exchange of octets of data between the two or more devices based on an Octet Stream Identifier which uniquely identifies each stream of data and related control information. The Octet Stream Identifier specifies the port or physical link with which the data and control information are associated as well as the byte position of the octet within the frame (if the transfer is one of framed data).

FIG. 1 illustrates schematically one embodiment of the invention in which a master and a slave are shown. The device which communicates with the PMD is described as the Octet Stream Master 20 since it provides the timing and sequencing for the transfers based on the availability of payload data from the facility and the requirement to provide payload data to the facility, e.g. TC sublayer. The device providing the TC sublayer is called the Octet Stream Slave 22 since it simply responds to requests made by the timing and sequence master. The direction of flow from the master to the slave is called the receive direction, and the direction opposite thereto is the transmit direction. The transmit and receive transfers both use the same transfer clock. If the maximum transfer clock is e.g., 50 MHz, the maximum transfer rate would be 100 Mbps in each direction. The transfer of data in blocks of a predetermined length, e.g. blocks of an octet, is via a common transfer clock and is synchronized at the block level.

Referring to FIG. 2, each exchange sequence consists of four transfer cycles and achieves transfer of a data block in two directions. In this embodiment, the data block is one octet long. However, it is, of course, possible to choose any reasonable length although, due to well adapted standards which presently exist, certain sizes would be more logical than others.

Referring further to the figure, the following signals are required:

ORD[11:0]

This is Octet stream Receive Data and contains up to 12 bits of data driven from the Octet Stream Master 20 to the Octet Stream Slave 22. ORD[11] is the MSB (most significant bit).

OTD[9:0]

This is Octet stream Transmit Data and contains up to 10 bits of data driven from the Octet Stream Slave 22 to the Octet Stream Master 20. OTD[9] is the MSB. OTD is tri-stated by all devices which are not programmed to respond to a particular TID value.

OCLK

This is Octet stream CLocK and is an octet transfer/synchronization clock for synchronizing transfers on both ORD and OTD.

OIS

This is Octet stream Interface Sync and is an active high signal asserted by the Octet Stream Master during the first transfer cycle of an exchange sequence.

Referring further to FIG. 2, operation and timing are described in detail in connection with clock cycles. The two unidirectional busses ORD and OTD are TDM busses which carry different information depending on the transfer cycle in the exchange sequence governed by OIS. An exchange sequence is the transfer of one complete set of parameters in both the transmit and receive directions.

During one exchange sequence, payload data blocks RxD and TxD identified by the master are exchanged in receive and transmit directions. The signalling information RxC and TxC so identified by the master are also exchanged in the same exchange sequence. The master identifies data and signalling information for exchange by sending the slave identification numbers in TID and RID.

According to the embodiment, an Octet Stream Identification number is a number from 0–1023 which is used to uniquely identify a specific octet stream. The Octet Stream Master applies a base Octet Stream Identifier for each link or virtual tributary supported. Consecutive Octet Stream Identifiers are then assigned for each channel in a link. If the link is unchannelized, the base octet identification number is used for all octets. The Octet Stream Slave device requires a programmable internal mapping of octet id's to channel numbers to provide a level of indirection.

Definition of the data content is as follows:

TID (Transmit IDentifier)

This is a 12 bit value sent by the Octet Stream Master device which identifies the required TxD and TxC in the next exchange sequence. TID[9:0] contains the Octet Stream Identifier number, TID[10] is asserted if data is required, and TID[11] is asserted if the octet requested is in the first frame of a multiframe, where a multiframe is service specific. An example of multiframe sizes is given in the "DS1/DS3 Example" on pages 8 and 9. Signalling bits will be provided by the Octet Stream Slave in the first frame in a multiframe if they are valid.

RID (Receive IDentifier)

This 12 bit value sent by the Octet Stream Master device identifies the RxD and RxC in the following two clock periods. RID[9:0] contains the Octet Stream Identifier number, RID[10] is asserted if valid data is present, and RID[11] is asserted if this octet is in the first frame of a multiframe. Valid signalling bits will be present with each octet in the first frame of a multiframe.

RxD (Receive Data)

This is an 8-bit payload value which is identified by RID on the previous clock cycle. This is valid if RID[10] is asserted.

TxD (Transmit Data)

This is an 8-bit payload value which is identified by the TID in the previous exchange sequence.

RxC (Receive Control)

This value contains control information passed from the Octet Stream Master to the Octet Stream Slave. This control information is identified by RID in two previous clock cycles. The contents of the control field are illustrated in the Table below when RxC is set for RCC=011.

	11	10:8	7	6	5	4	3	2	1	0
RxControl	LS	RCC	A	B	C/A'	D/B'	RTS			

TxC (Transmit Control)

This value contains control information passed from the Octet Stream Slave to the Octet Stream Master. This control information is identified by the TID in the previous block exchange sequence. The contents of the control field are illustrated in the Table below when TxC is set for TCC=11.

	9:8	7	6	5	4	3	2	1	0
TxControl	TCC	A	B	C/A'	D/B'	RTS			

RCC=Receive Control Code. This code indicates what data will be present in bits 7-0 as follows:

000: RXC[7:0]=User defined field
001: RXC[7:4]=Updated Signalling bits
: RXC[3:0]=Unused
010: RXC[7:4]=Unused
: RXC[3:0]=Updated SRTS value
011: RXC[7:4]=Updated Signalling bits
: RXC[3:0]=Updated SRTS value
100–111: RXC[7:0]=User defined field

TCC=Transmit Control Code. This code indicates what data will be present in bits 7-0 as follows:

00: TXC[7:0]=User defined field
01: TXC[7:4]=Updated Signalling bits
: TXC[3:0]=Unused
10: RXC[7:4]=Unused
: RXC[3:0]=Updated SRTS value
11: RXC[7:4]=Updated Signalling bits
: RXC[3:0]=Updated SRTS value

ABCD=ABCD Signalling bits

ABA'B'=AB Signalling bits for two superframes when using DS1 SF format. A'B' are the signalling bits for odd numbered superframes and AB is for even numbered superframes.

LS=Link Status for link corresponding to the RID. This is a user defined status field used to pass link information such as LOS, OFF, LOF, etc. as required.

RTS=Rx RTS value. This is CES specific timing information, as defined in the I363.1 specification. This is an RTS formatted value used to support SRTS or any other clock recovery mechanism. Bit 3=RTS most significant bit.

In the embodiment where a single DS3 master to a single slave is used, the following identification numbers should be used in non-transparent mode.

	DS1 #1	DS1 #2	DS1 #3	...	DS1 #28
DS0 #1	0	24	48	...	648
DS0 #2	1	25	49	...	649

-continued

	DS1 #1	DS1 #2	DS1 #3	...	DS1 #28
DS0 #3	2	26	50	...	650
.	.	.	.	...	.
.	.	.	.	...	.
.	.	.	.	...	.
DS0 #24	23	47	71	...	671

In the transparent mode, the following numbers apply:

	DS1 #1	DS1 #2	DS1 #3	...	DS1 #28
DS0 #1	0	24	48	...	648
DS0 #2	0	24	48	...	648
DS0 #3	0	24	48	...	648
.	.	.	.	...	.
.	.	.	.	...	.
.	.	.	.	...	.
DS0 #24	0	24	48	...	648

It should be noted that the DS1–DS3 embodiments are given as example only. It would be apparent to those skilled in the art that the invention is equally applicable to other digital signal streams, such as E1, J2, STS-1, E3, STM-1, etc.

Referring to FIG. 3, another embodiment is schematically illustrated in which multiple slaves are provided with one master. If multiple masters are required, they must be connected in a manner to appear as one Octet Stream Master as defined herein.

One other preferred embodiment is schematically illustrated in FIG. 4 in which, as in FIG. 3, multiple slaves are provided with one master. The master in this embodiment includes a M13 multiplexer for handling DS3 signals in the physical medium sublayer. The master includes 28 port T1 framers which are connected to three slaves by busses and other connections. ORD and OTD are time division multiplexed busses. On the transmission convergence sublayer, one slave carries up to 672 channel HDLC (high level data link control) signals, another up to 28 ATM TC sublayer signals. The third slave handles up to 672 channel AAL1 (ATM adaptation layer 1) signals. Of course, other TC sublayer signals are possible.

What is claimed is:

1. A method of synchronously transferring data and other information between a physical medium sublayer and a transmission convergence sublayer, each handling a non-synchronized digital signal stream, comprising steps of:

a master bidirectionally handling a non-synchronized digital signal stream to/from the physical medium sublayer;

a slave bidirectionally handling a non-synchronized digital signal stream to/from the transmission convergence sublayer;

the slave and the master exchanging the data and other information in individually identifiable blocks of predetermined lengths during each successive exchange sequence of a preset length of time in a time synchronized fashion under a clock signal;

the method further comprising steps of:

during each exchange sequence,

the master sending the slave identifiers of blocks of data and other information to be exchanged between the slave and the master; and

in response to the identifiers, the slave exchanging with the master the blocks of data and other information so identified.

2. The method according to claim 1 wherein an exchange sequence comprises a plurality of transfer cycles, and during each transfer cycle, the master sends the slave, respectively, the identifier of the data block to be received from the slave during the next exchange sequence, a data block, other information and their identifier.

3. The method according to claim 2 wherein during the coincidental transfer cycles, the slave sends the master respectively the data block and other information so identified by the master in the previous exchange sequence.

4. The method according to claim 3 wherein the other information exchanged between the master and the slave comprises control signals which include a timing signal, signalling information and/or other user definable information.

5. The method according to claim 4 wherein the timing signal is RTS values.

6. The method according to claim 4 wherein the signalling information is exchanged by the use of standard signalling bits contained in the control signals.

7. The method according to claim 4 wherein there is more than one slave and the method comprises a further step of:

one of the slaves, in response to the identifiers, exchanging with the master the blocks of data and other information so identified.

8. The method according to claim 7 wherein the non synchronized digital signal streams are any of DS1, E1, J2, DS3, E3, STS-1, and STM-1.

9. An interface for synchronously transferring data and other information between a physical medium sublayer and a transmission convergence sublayer, each sublayer handling a non-synchronized digital signal stream, comprising:

a master for bidirectionally handling a non-synchronized digital signal stream to/from the physical medium sublayer;

a slave for bidirectionally handling a non-synchronized digital signal stream to/from the transmission convergence sublayer under the control of the master;

time division multiplexed buses connecting the master and the slave for exchanging the data and other information so identified by the master in individually identifiable blocks of predetermined lengths during each successive exchange sequence of a preset length of time in a time synchronized fashion under a same clock signal; and

the identifiable blocks in each successive exchange sequence containing data and their identifiers.

10. The interface according to claim 9, wherein the identifiable blocks in one successive exchange sequence on one time division multiplexed bus contain data and control signals and their identifier in addition to an identifier of data to be transferred on the second time division multiplexed bus in the following exchange sequence.

11. The interface according to claim 10, wherein the identifiable blocks in one successive exchange sequence on another time division multiplexed bus contain data and control signals so identified by an identifier on the first time division multiplexed bus in the previous exchange sequence.

12. The interface according to claim 11, wherein each exchange sequence is divided into four transfer cycles and each identifiable block is transferred in each transfer cycle in one direction over one time division multiplexed bus.

13. The interface according to claim 12, wherein there is more than one slave.

* * * * *

US005241543A

United States Patent [19]

Amada et al.

[11] Patent Number: 5,241,543

[45] Date of Patent: Aug. 31, 1993

[54] INDEPENDENT CLOCKING LOCAL AREA NETWORK AND NODES USED FOR THE SAME

[75] Inventors: Eiichi Amada, Tokyo; Kunio Hiyama, Fujisawa; Naoya Kobayashi, Hachioji; Yoshihiro Takiyasu, Higashimurayama; Yasuhiko Hatekeyama, Hadano; Haruyuki Nakayama, Ebina, all of Japan

[73] Assignee: Hitachi, Ltd., Tokyo, Japan

[21] Appl. No.: 672,640

[22] Filed: Mar. 20, 1991

Related U.S. Application Data

[63] Continuation-in-part of Ser. No. 399,901, Aug. 29, 1989, Pat. No. 5,103,447.

[30] Foreign Application Priority Data

Jan. 25, 1989 [JP] Japan 1-013910
Mar. 22, 1990 [JP] Japan 2-069779

[51] Int. Cl.[5] H04L 7/00

[52] U.S. Cl. 370/100.1; 370/102; 370/85.15

[58] Field of Search 370/100.1, 102, 105, 370/105.1, 105.2, 105.3, 105.4, 106, 85.15, 85.12, 85.5; 375/107, 114, 116, 118, 120

[56] References Cited

U.S. PATENT DOCUMENTS

4,569,041 2/1986 Takeuchi et al. 370/85.15
4,791,652 12/1988 McEachern et al. 370/102
4,811,340 3/1989 McEachern et al. 370/102
4,858,232 8/1989 Diaz et al. 370/85.15

FOREIGN PATENT DOCUMENTS

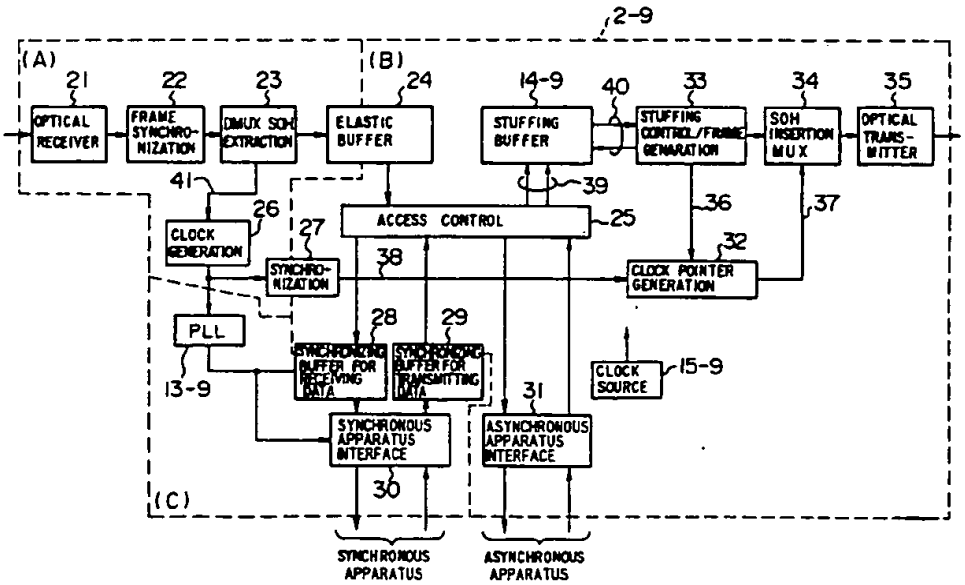
6144426 7/1981 Japan .

Primary Examiner—Douglas W. Olms
Assistant Examiner—Hassan Kizou
Attorney, Agent, or Firm—Antonelli Terry Stout & Kraus

[57] ABSTRACT

In a local area network composed of transmission lines for interconnecting a plurality of subordinate networks including synchronous apparatuses and a plurality of nodes which connect the subordinate networks to the transmission lines, information is transferred using a fixed length frame, a clock source which generates an independent clock signal and a circuit which generates a fixed length frame with the oscillation frequency of the clock source as a reference are provided in each node so as to adopt an independent clocking system, and distribution of a common synchronizing clock required for synchronous apparatuses is made by transmission while embedding transition point information of a synchronizing clock in a specific space in the fixed length frame. Further, each node generates a fixed length transmission frame with an independent clock signal, and on the other hand, there are provided in each node, from the requirement that information quantity applied to a network is made constant, a circuit for extracting a received clock, a storage device for storing received information temporarily, and an information outgoing quantity control circuit in which information quantity which is sent out in one frame is increased when the information quantity stored in the storage device becomes more than a predetermined first reference value and information quantity which is sent out in one frame is decreased when the information quantity stored in the storage device becomes less than a predetermined second reference value.

18 Claims, 10 Drawing Sheets



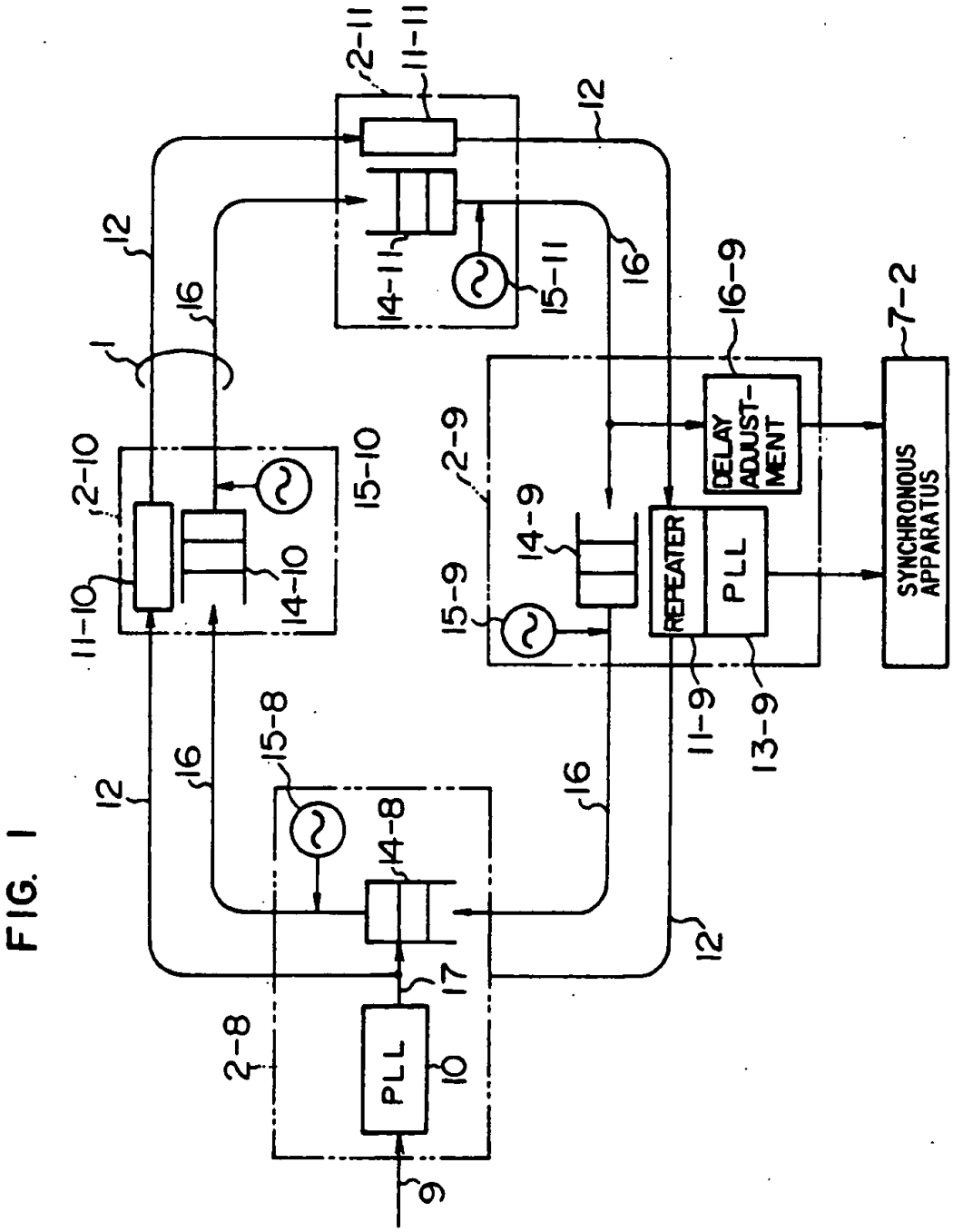

2-11
11-11
12
14-11
15-11
16
16-9
7-2
DELAY ADJUST-MENT
SYNCHRONOUS APPARATUS
2-9
14-9
15-9
REPEATER
PLL
11-9
13-9
2-10
15-10
11-10
14-10
15-8
14-8
17
2-8
PLL
10
9

FIG. 1

FIG. 2

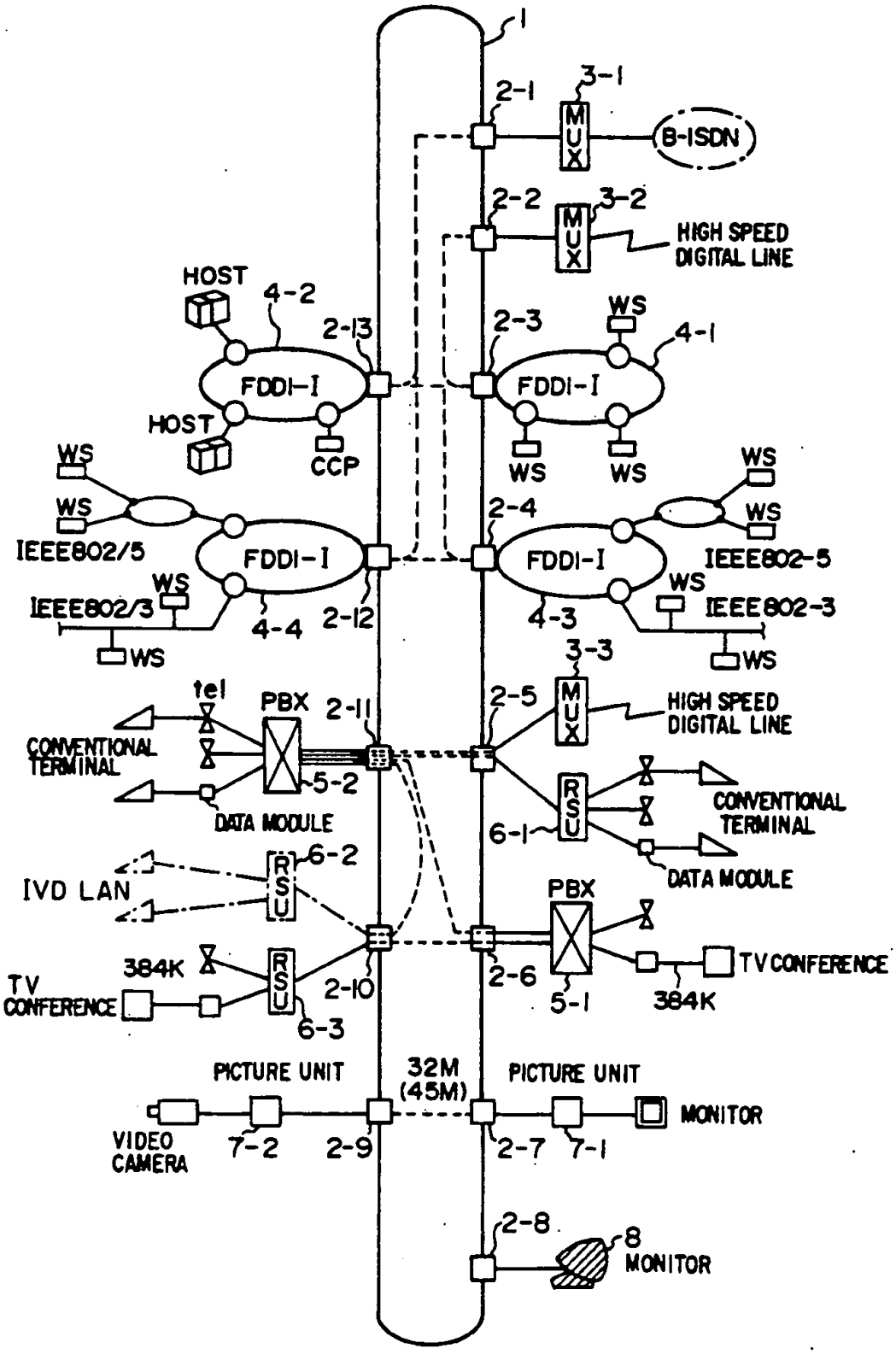

1
2-1
3-1
MUX
B-ISDN
2-2
3-2
MUX
HIGH SPEED DIGITAL LINE
HOST
4-2
2-13
2-3
WS
4-1
FDDI-I
FDDI-I
HOST
CCP
WS
WS
WS
WS
WS
WS
2-4
IEEE802/5
FDDI-I
FDDI-I
IEEE802-5
WS
WS
IEEE802/3
WS
4-4
2-12
4-3
IEEE802-3
WS
WS
3-3
tel
PBX
2-11
2-5
MUX
HIGH SPEED DIGITAL LINE
CONVENTIONAL TERMINAL
5-2
DATA MODULE
RSU
6-1
CONVENTIONAL TERMINAL
DATA MODULE
6-2
IVD LAN
RSU
PBX
384K
RSU
2-10
2-6
TV CONFERENCE
TV CONFERENCE
6-3
5-1
384K
PICTURE UNIT
32M (45M)
PICTURE UNIT
MONITOR
VIDEO CAMERA
7-2
2-9
2-7
7-1
2-8
8
MONITOR

FIG. 3

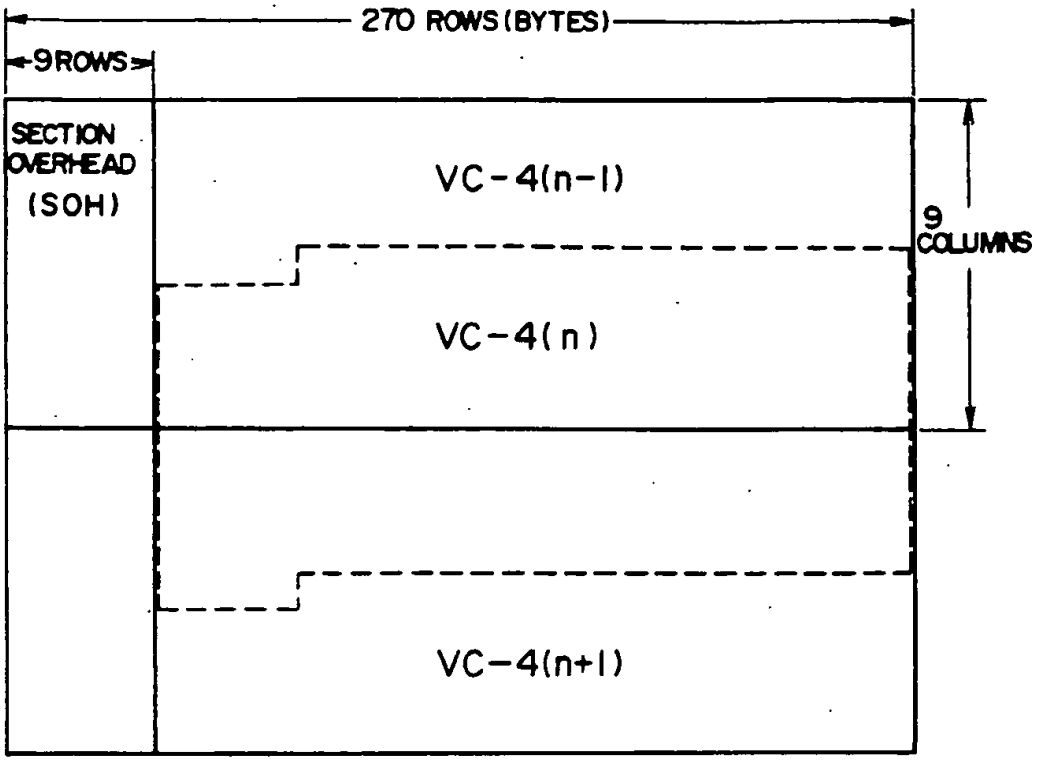
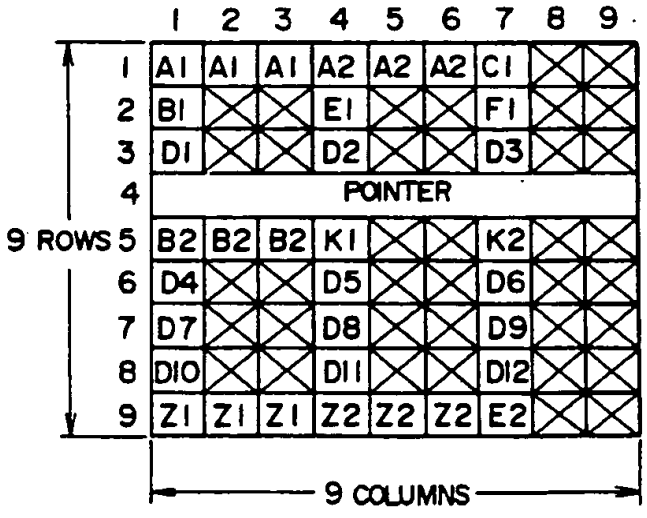

270 ROWS (BYTES)
9 ROWS
SECTION OVERHEAD (SOH)
VC-4(n-1)
VC-4(n)
VC-4(n+1)
9 COLUMNS

FIG. 4


1 2 3 4 5 6 7 8 9
1 A1 A1 A1 A2 A2 A2 C1
2 B1 E1 F1
3 D1 D2 D3
4 POINTER
9 ROWS 5 B2 B2 B2 K1 K2
6 D4 D5 D6
7 D7 D8 D9
8 D10 D11 D12
9 Z1 Z1 Z1 Z2 Z2 Z2 E2
9 COLUMNS

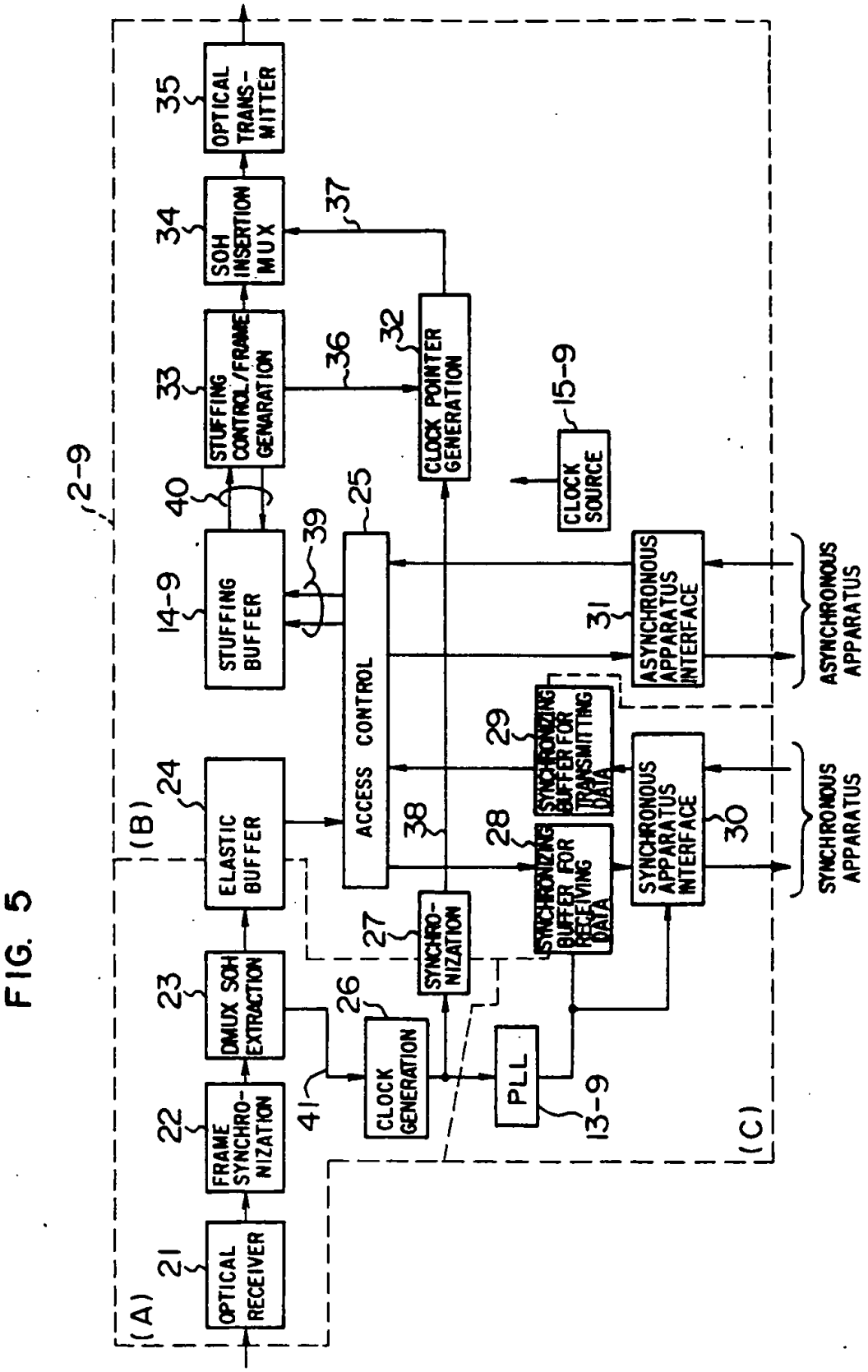

OPTICAL TRANSMITTER
35
SOH INSERTION MUX
34
37
STUFFING CONTROL/FRAME GENARATION
33
36
CLOCK POINTER GENERATION
32
15-9
CLOCK SOURCE
2-9
40
39
25
STUFFING BUFFER
14-9
ACCESS CONTROL
ASYNCHRONOUS APPARATUS INTERFACE
31
ASYNCHRONOUS APPARATUS
29
SYNCHRONIZING BUFFER FOR TRANSMITTING DATA
(B)
24
ELASTIC BUFFER
38
28
SYNCHRONIZING BUFFER FOR RECEIVING DATA
SYNCHRONOUS APPARATUS INTERFACE
30
SYNCHRONOUS APPARATUS
27
SYNCHRO-NIZATION
23
DMUX SOH EXTRACTION
26
CLOCK GENERATION
PLL
22
FRAME SYNCHRO-NIZATION
41
13-9
(A)
21
OPTICAL RECEIVER
(C)

FIG. 5

FIG. 6

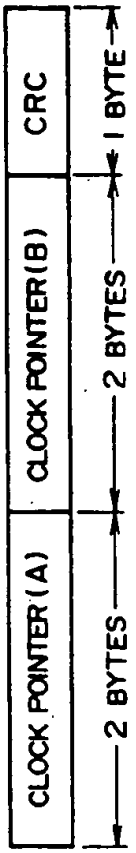

CLOCK POINTER(A)
CLOCK POINTER(B)
CRC
2 BYTES
2 BYTES
1 BYTE

FIG. 7

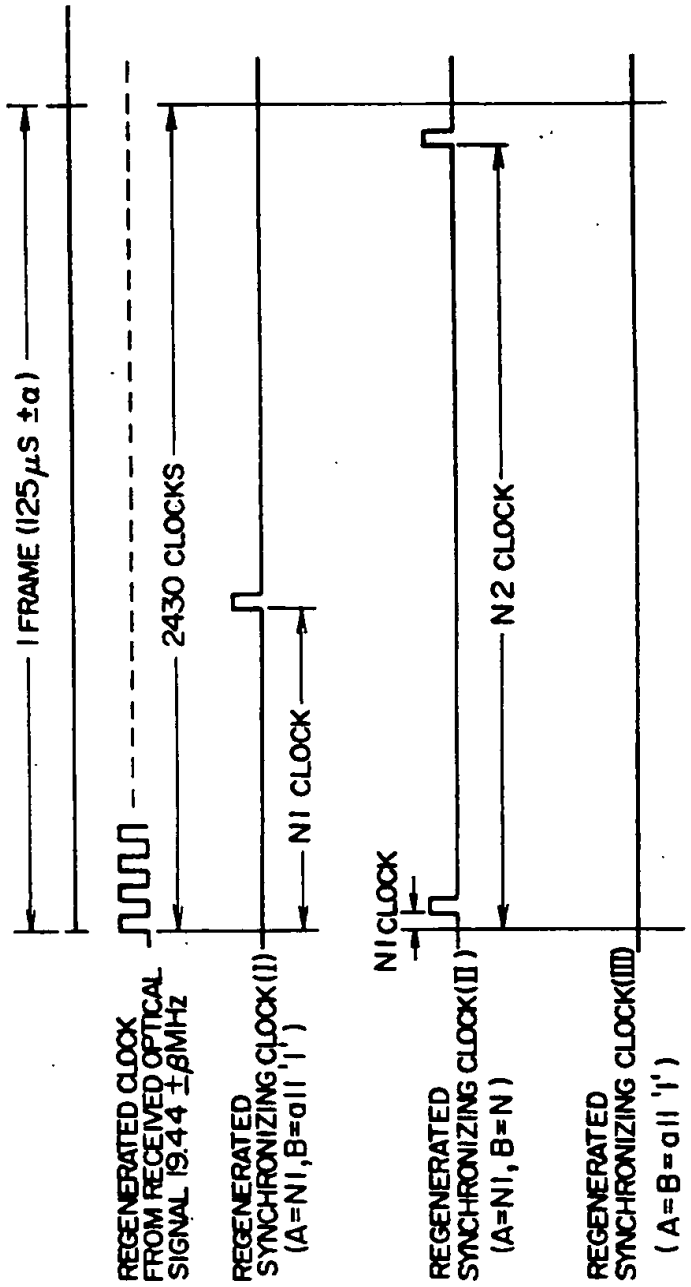

1 FRAME (125μS ±α)
2430 CLOCKS
N1 CLOCK
N1 CLOCK
N2 CLOCK
REGENERATED CLOCK FROM RECEIVED OPTICAL SIGNAL 19.44 ±βMHz
REGENERATED SYNCHRONIZING CLOCK(I) (A=N1, B=all '1')
REGENERATED SYNCHRONIZING CLOCK(II) (A=N1, B=N)
REGENERATED SYNCHRONIZING CLOCK(III) (A=B=all '1')

FIG. 8.

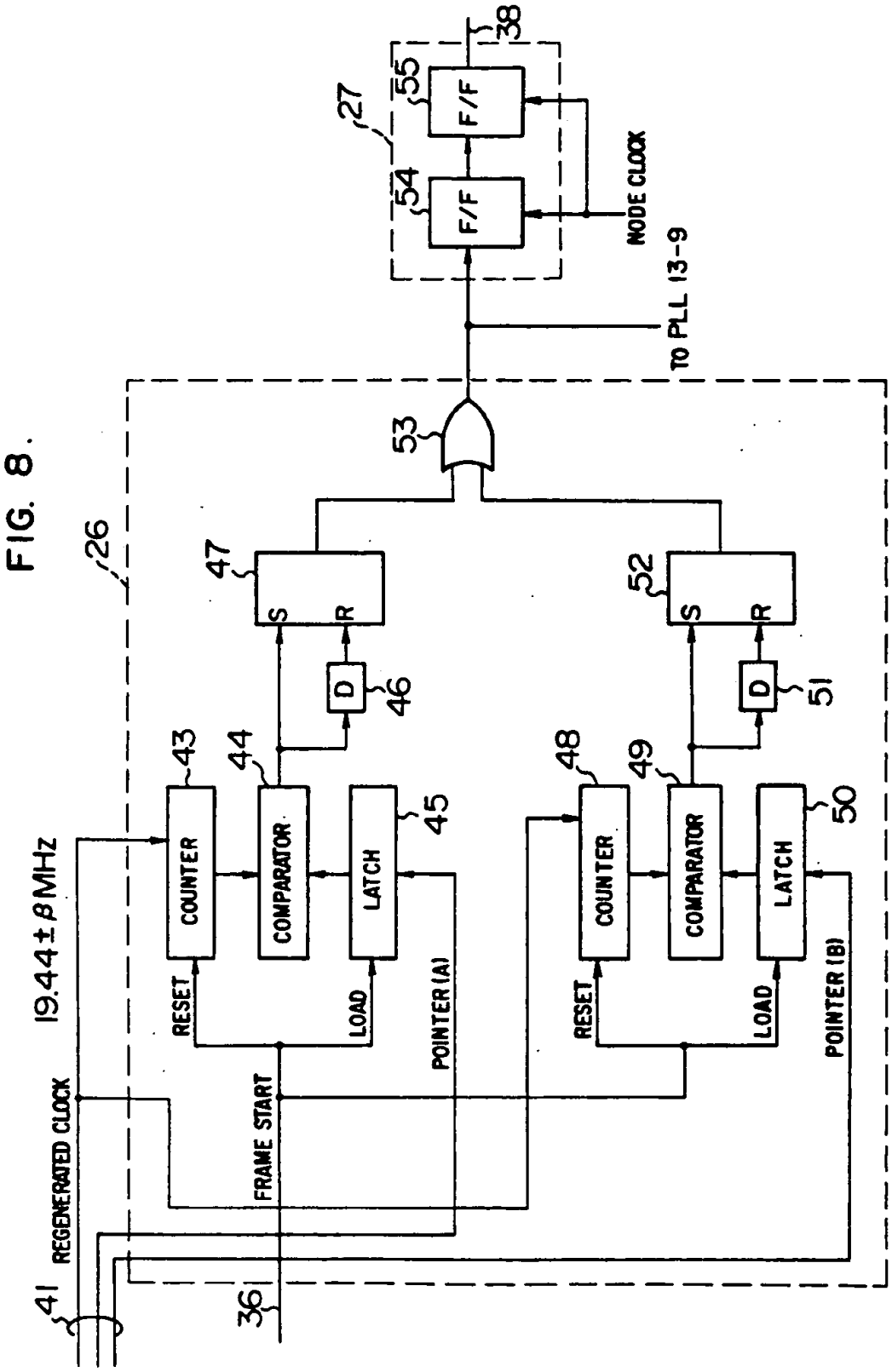

38
27
55
F/F
54
F/F
MODE CLOCK
TO PLL 13-9
53
26
47
S
R
52
S
R
D
46
D
51
43
44
48
49
COUNTER
COMPARATOR
LATCH
45
COUNTER
COMPARATOR
LATCH
50
19.44±βMHz
RESET
LOAD
POINTER (A)
RESET
LOAD
POINTER (B)
REGENERATED CLOCK
FRAME START
41
36

FIG. 9

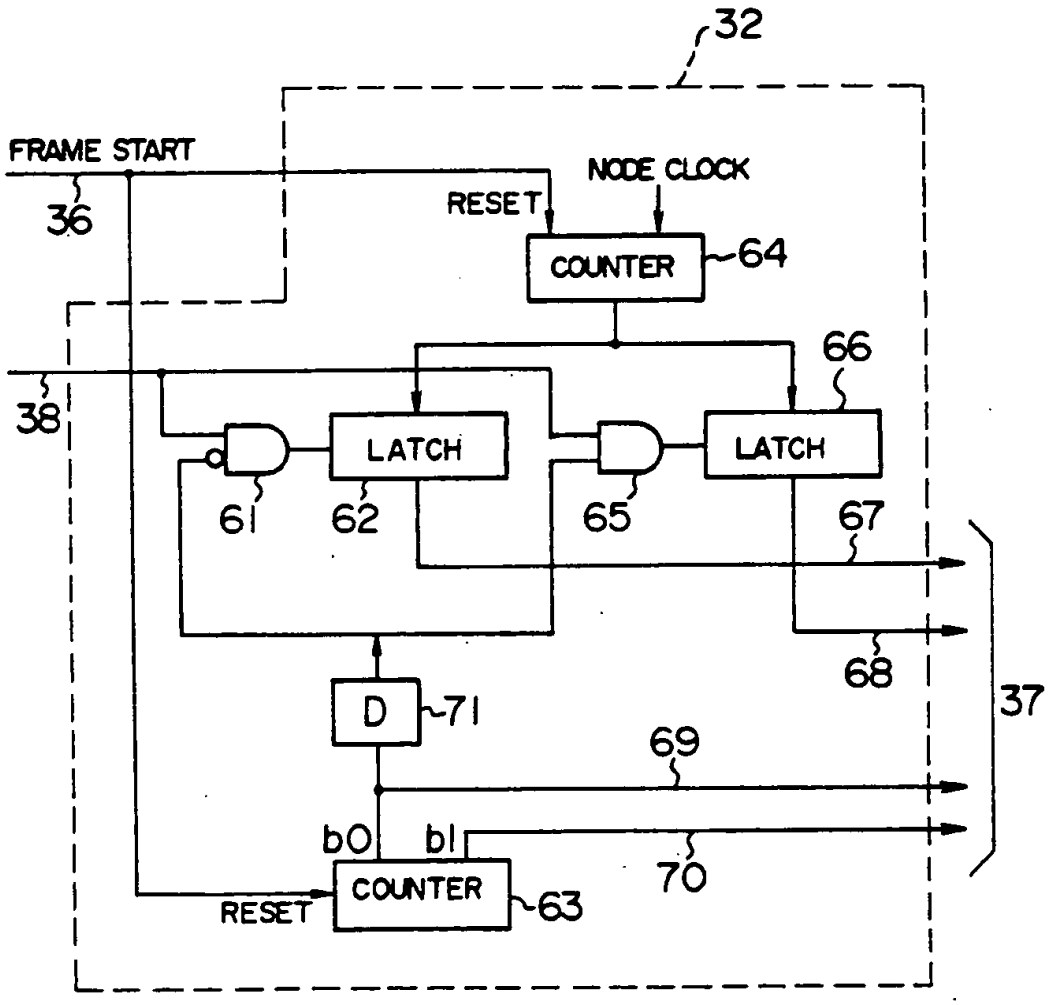

32
FRAME START
36
NODE CLOCK
RESET
COUNTER
64
66
38
LATCH
LATCH
61
62
65
67
68
D
71
37
69
b0
b1
70
RESET
COUNTER
63

FIG. 10

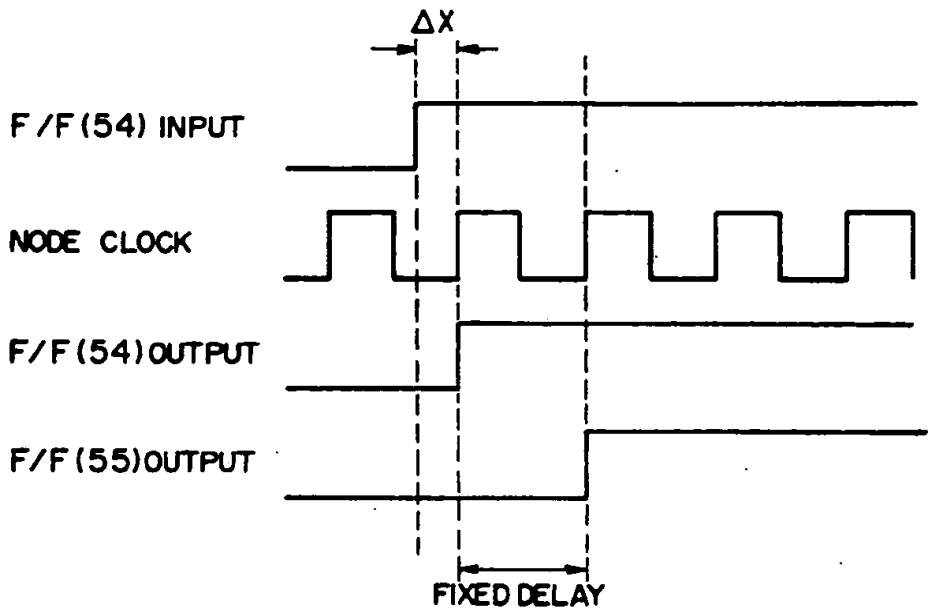

ΔX
F/F(54) INPUT
NODE CLOCK
F/F(54)OUTPUT
F/F(55)OUTPUT
FIXED DELAY

FIG. 11

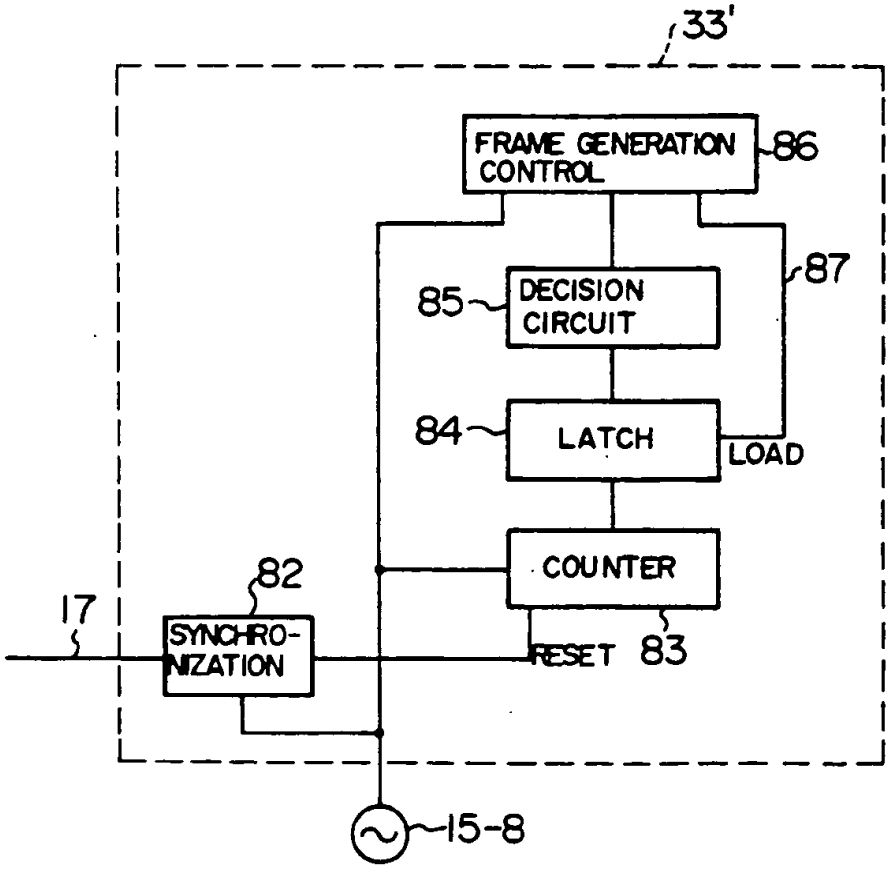

33'
FRAME GENERATION CONTROL
86
87
85
DECISION CIRCUIT
84
LATCH
LOAD
COUNTER
82
17
SYNCHRO-NIZATION
RESET
83
15-8

FIG. 12

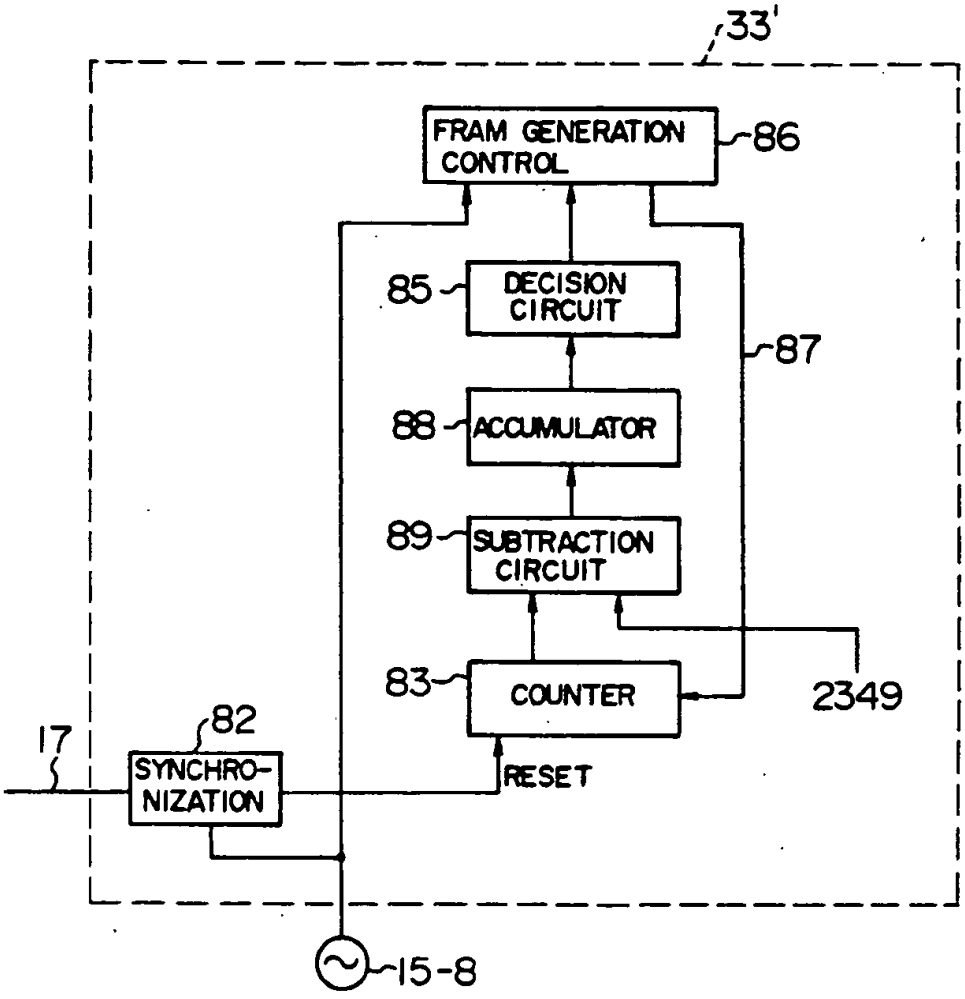

33'
FRAM GENERATION CONTROL
86
85
DECISION CIRCUIT
87
88
ACCUMULATOR
89
SUBTRACTION CIRCUIT
83
COUNTER
2349
82
17
SYNCHRO-NIZATION
RESET
15-8

FIG. 13

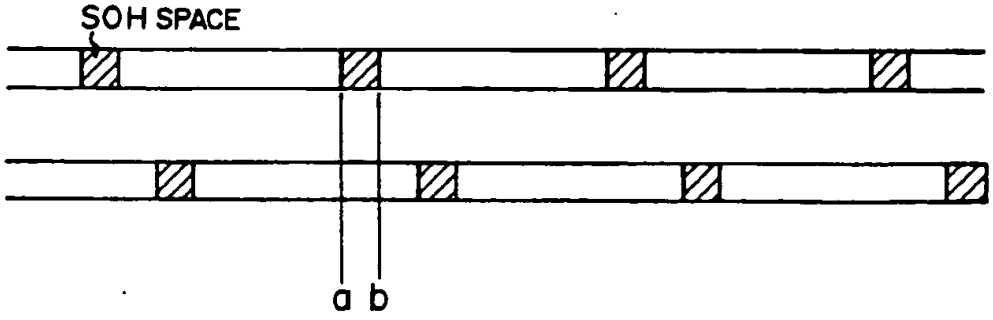

SOH SPACE
a
b

FIG. 14

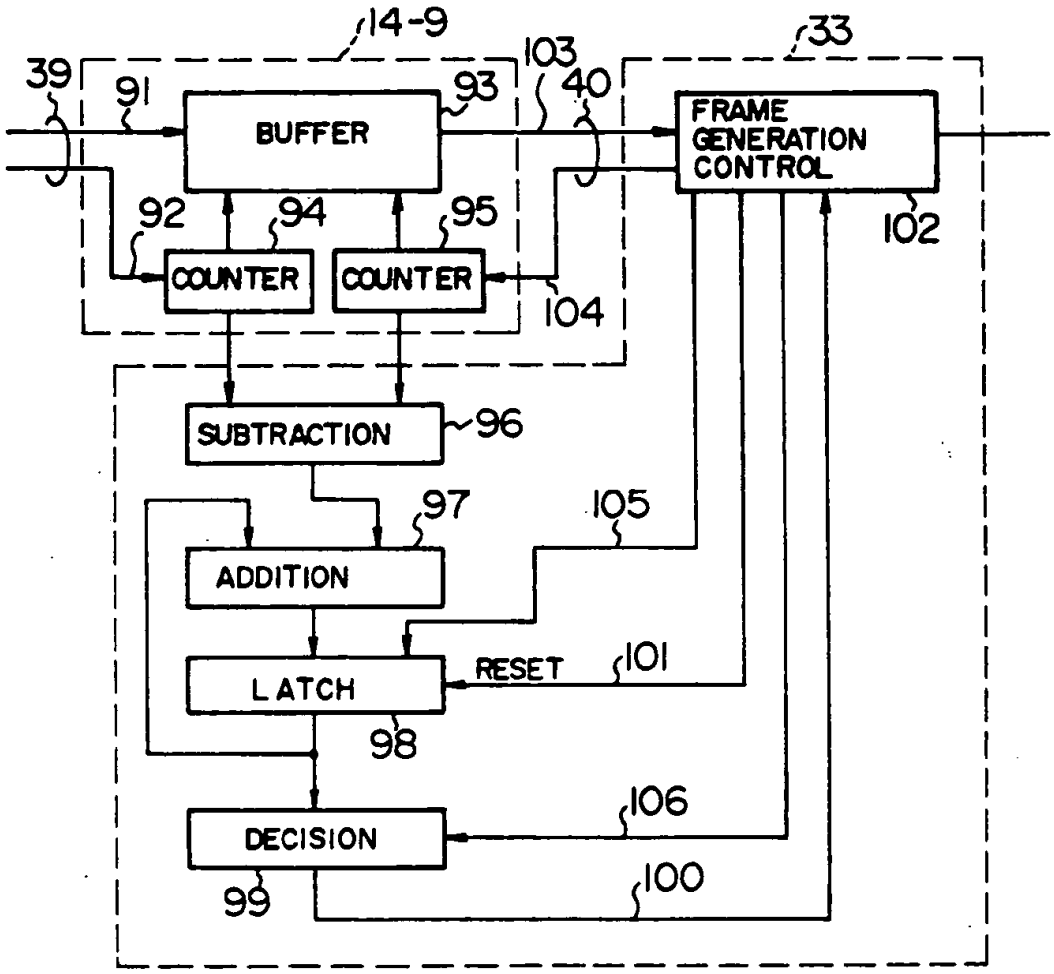
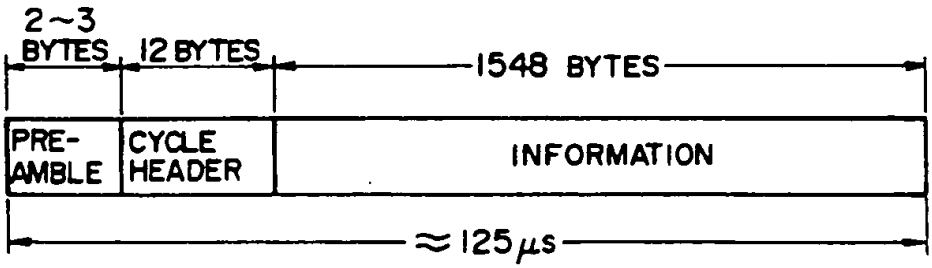

14-9
33
39
91
BUFFER
93
103
40
FRAME GENERATION CONTROL
92
94
95
102
COUNTER
COUNTER
104
SUBTRACTION
96
97
105
ADDITION
LATCH
RESET
101
98
106
DECISION
99
100

FIG. 15


2~3 BYTES
12 BYTES
1548 BYTES
PRE-AMBLE
CYCLE HEADER
INFORMATION
≈125μs

INDEPENDENT CLOCKING LOCAL AREA NETWORK AND NODES USED FOR THE SAME

CROSS-REFERENCE TO RELATED APPLICATION

This is a continuation-in-part of copending U.S. application Ser. No. 07/399,901, filed Aug. 29, 1989 which issued as U.S. Pat. No. 5,103,447.

BACKGROUND OF THE INVENTION

The present invention relates to an architecture of an independent clocking local area network (LAN), i.e., a local area network (hereinafter abbreviated as a LAN) in which each node has an independent clock source of a clock signal and an information signal is sent out using an oscillated clock signal, and more particularly, to an architecture of a multimedia LAN in which any data transfer error due to jitter accumulation is not generated even when the quantity of connected nodes is increased.

In a LAN, a plurality of node devices (hereinafter simply referred to as nodes) are connected with one another with a single transmission line and high speed information transmission and switching function are realized efficiently in a limited service area, and variety types of LANs have been put to practical use. A ring type LAN having a transmission line in a ring form and a bus type LAN using a transmission line in a bus form are typical. In such a LAN construction, a synchronous system becomes an issue. Ideally, it is preferable that all the nodes constituting a LAN are operated with a same clock signal (hereinafter simply referred to as a clock). In case all the nodes are in operation with the same clock, the rate of sending information and the rate of receiving information are equal to each other. Therefore, transmission and reception of information become possible without providing a buffer therebetween. As such a LAN in which all the nodes are operated with the same clock, the standard IEEE 802.5 ("Token ring") (a document ANSI/IEEE Std 802.5-1985 ISO/DP 8802/5 "LOCAL AREA NETWORKS Token Ring Access Method") is a typical well-known example. In respective nodes in a LAN of above-mentioned standard IEEE 802.5, clock components included in a received signal from a previous (transmission) node are regenerated by means of a phase lock loop (PLL), regenerated clock is supplied into a receiving node, and further, information is sent out to a next node by abovementioned regenerated clock (master-slave synchronization). The clock for synchronization is repeated at respective nodes as described above, makes a round in the ring, and the whole system becomes to be operated synchronously with a synchronous clock generated by a master node. Since the clock is regenerated and repeated in respective nodes, however, the jitter generated at the time of regeneration and transfer of the clock is accumulated. Since received data are regenerated by the clock having such jitter, such a problem arises that received data are not regenerated correctly when the jitter becomes larger. The quality of connectable nodes is limited in many cases in point of operation by such jitter accumulation.

In order to avoid jitter accumulation, there is an (independent synchronizing) system in which regeneration and transfer of the clock is not performed, each of respective nodes has an independent clock source, respectively, and an information signal is sent out using an oscillated clock. For example, this system is described in detail in the standard FDDI-I (a document ISO/IEC JTCl SC13 N477; Draft for ISO 9314-1: Fiber Distributed Data Interface (FDDI) Token Ring Physical Layer Protocol (PHY)).

In FDDI-I, however, the information signal transmitted in the LAN is asynchronous information, viz., only the information which is not required to be sent periodically, and the information is transmitted and received with a frame for sending information (hereinafter referred to as a frame) (4,500 bytes maximum) as a unit. A blank of 8 bytes and more is put between mutual frames, and the difference in clock frequencies between nodes is absorbed by increasing and decreasing the size of the blank portion. Thus, it is possible for respective nodes to conduct communication without giving rise to overflow or underflow by regenerating and repeating data only.

The above-mentioned independent clocking system is a system which is applicable only to a LAN which supports asynchronous data only. Recently, however, demand for a high speed LAN called a multimedia backbone LAN which is able to transmit and switch not only asynchronous data, but also synchronous information is increasing. (Information, voice and data which are required to transmit a predetermined quantity periodically are typical examples. These may be handled as asynchronous information, but buffering processing and the like are required to guarantee periodicity at transmit-receive terminals, causing handling to become complicated.) Such a multimedia backbone LAN accommodates a low speed, asynchronous-data-dedicated LAN such as the standard IEEE 802.3, 802.4 and 802.5 and FDDI-I which is a high speed LAN so as to realize information transmission and switching function among LANs, and also supports information transfer among synchronous apparatuses such as a PBX (private branch exchange) and a TDM (time division multiplexer) so as to realize an integrated private network. Existing synchronous apparatuses are designed on the premise that these apparatuses are operated with the same synchronizing clock when they are interconnected. Accordingly, in a network including such synchronous apparatuses it is required to supply a synchronizing clock to synchronous apparatuses from the network through nodes. Further, since it is required to transfer information periodically and at a same rate among synchronous apparatuses, it is preferable that an information quantity applied to respective nodes is made equal in the whole system. Thus, it is required to supply a synchronizing clock which is common to all nodes.

As a result, a master-slave synchronization system which is easy to be constructed has been heretofore employed for synchronization of the multimedia LAN. As a document related to such a technique, "A 1.2 Gbps optical loop LAN for wideband office communications" IEEE Global Telecommunications Conference 1985, 15-4, may be mentioned.

In the above-mentioned LAN of master-slave synchronization system, the synchronizing clock is distributed by the fact that the clock generated by a master node is regenerated and repeated by respective nodes. In this system, since all nodes are operated with a common synchronizing clock, it is easy to connect synchronous apparatuses with one another. Since jitter is accumulated as described previously, however, there is such

a drawback that the number of connectable nodes is limited.

As another system for solving the jitter accumulation problem in the multimedia LAN, an independent clocking system in which respective nodes send out signals to a transmission line using clocks oscillated in respective stations is possible. In the multimedia LAN, however, it is necessary to devise how to include the synchronous apparatuses, unlike the synchronous-data-dedicated LAN. For example, it is being examined to employ an independent clocking system in the standard FDDI-II which is being standardized by the American National Standards Institute (ANSI) at present described in detail in a document: "FDDI Hybrid Ring Control, Draft proposed American Standard, Jan. 20, 1989". FIG. 15 shows a construction of a transfer frame (referred to as a cycle in FDDI-II) adopted in FDDI-II. The information is transferred while being embedded in a transfer frame of a fixed period. The frame is composed of a preamble, a cycle header and an information portion. The period of the frame is at 125 μs (8 KHz). Further, the information transmission rate is at 100 Mb/s, but information in 4 bits is sent out after converting into 5 bits (4B/5B code) for the purpose of removing DC frequency components on a transmission line and transmission of specific codes (for detection of frame boundary and control signals). Therefore, the physical transmission rate is at 125 Mb/s. The number of bits in the preamble space is different depending on oscillation frequency deviation of clocks of respective nodes, but the number of bits is adjusted so that the cycle period becomes 125 μs. The master node creates the frame period based on an external clock or an oscillation frequency of the own station. In each node, a synchronizing clock is extracted from a received signal using a PLL, a tank circuit and the like. It becomes possible to receive information in a frame by receiving the received signal correctly and detecting a synchronous pattern in the cycle header using the extracted clock.

In a proposal in the above-mentioned standard of FDDI-II, the oscillation frequency deviation of each node is adjusted by adjusting the length of the preamble portion between frames, and periodic data transfer is realized by introducing a frame construction.

It is required for a multimedia LAN to distribute the same synchronizing clock among synchronous apparatuses through the nodes in order to transmit not only asynchronous information, but also synchronous information as described previously. Accordingly, there is a problem as a synchronous system in both systems of the above-mentioned master-slave synchronization system and FDDI-II system of independent clocking. That is, restriction on the number of nodes due to jitter accumulation described previously becomes an issue in the master-slave synchronization system.

In the FDDI-II system, there are such problems as described hereunder.

A first problem is that the system is weak against a transmission error on the transmission line. In a high speed LAN, an optical fiber is used for transmission, but a bit error rate in optical transmission is usually around 10^{-9}. Such a bit error generated at random or in a burst form should never be enlarged by the network. In the FDDI-II system, a starting point of each frame is recognized by detecting a specific bit pattern which does not exist in the information, and there is a possibility that an error of one frame portion is generated by the bit error at this portion. Further, the length of an outputted frame is determined by the length of a received frame in each node, and there is also a possibility that a frame recognition error of one node extends to a plurality of nodes.

A second problem exists in that a physical transmission rate becomes higher than a logical information transfer rate. This is caused by the fact that 4-bit information is transmitted after coding into a 5-bit transmission code because a specific bit pattern which does not appear in the information portion is used for frame recognition. In FDDI-II, the physical transmission rate is set at 125 Mb/s against the information transfer speed of 100 Mb/s, and only 80% of the transmission band is utilized for actual information transfer.

A third problem is that frame processing becomes complicated because a frame is of a variable length.

SUMMARY OF THE INVENTION

Thus, it is an object of the present invention to realize an independent local area network and nodes for a local area network which have solved above-mentioned problems, that is, which have little jitter accumulation and are able to distribute the synchronizing clock among synchronous apparatuses even when a frame having a fixed length is used.

In order to achieve the above-mentioned object, according to the present invention, there is provided a local area network composed of transmission lines for interconnecting a plurality of subordinate networks including synchronous apparatuses and a plurality of nodes which connect the above-mentioned subordinate networks to the transmission lines, wherein information is transferred using a fixed length frame, a clock source generating an independent clock signal in each node and means for forming a fixed length frame with the oscillation frequency of the clock source as a reference are provided whereby to adopt an independent clocking system, and distribution of a common synchronizing clock required for synchronous apparatuses is made in such a manner that transition point information of the synchronizing clock is embedded in a specific place in the fixed length frame in transmission. The above-mentioned transition point information means that reference points such as rising or falling edges of the clock provide timewise positional information during the period of a transmission frame having a fixed length formed by each of the above-mentioned nodes. Since each node has an independent clock, transition point information is varied with respect to each node because the period of a fixed length transmission frame formed by each node and the period of a common synchronizing clock are independent.

Further, from such requirements that each node produces a fixed length transmission frame with an independent clock signal, and on the other hand, the information quantity applied to a network is made constant, there are provided in each node, means of extracting a received clock, storage means for storing received information temporarily, and information outgoing quantity control means which increases information quantity which is sent out into one frame when the information quantity stored in the storage means becomes more than a predetermined first reference value and reduces the information quantity which is sent out into one frame when the information quantity stored in the storage means becomes less than a predetermined second reference value.

The common synchronizing clock is sent out from the master node as a preferable embodiment configuration. Further, NNI ("Network Node Interface for the Synchronous Digital Interface") standards which are specified by CCIT (International Telegraph and Telephone Consultative Committee) standards are applied to a physical layer. In particular, a SONET (Synchronous Optical Network) frame is used as the fixed length frame, and the transition point information on the common synchronizing clock is transferred by using a section overhead space of the SONET frame.

The clock frequency of each node is independent (independent clocking), and the rate of transmitted information is common in the whole system. Accordingly, a stuffing function of NNI standard is used for absorbing the difference between the node clock frequency and the information transmission rate in each node. The synchronizing clock is distributed by setting the overhead portion of NNI standard, viz., the synchronizing clock period to be distributed at almost the same frequency as the frame frequency, and by transferring transition point information of the synchronizing clock.

According to the present invention, the problem of data transmission error caused by clock jitter accumulation is solved by making the clock frequency in each node independent. Furthermore, connectability with public networks and appropriation of techniques become possible by applying NNI standards which are international standards to the physical layer. Further, frame synchronization is obtainable by using a fixed length frame even if a specific pattern for frame synchronization is not used. Namely, since a frame synchronous pattern appears periodically even if the same pattern as the pattern for frame synchronization is used in the information portion, it is possible to recognize a starting point of a frame by detecting periodicity. Thus, it is possible to make the physical transmission rate and the information transmission rate almost equal to each other. Further, independent synchronization can be realized using the stuffing function of NNI standard. Namely, in each node, the difference between a self-node clock and the information quantity from a previous node is monitored, and stuffing is performed in a direction of reducing the difference when the difference exceeds a specified threshold, whereby making it possible to make the information quantity applied to transmission lines constant while maintaining the clock frequency in each node independent. The information rate applied to the transmission line is specified by the master node. Furthermore, distribution of common synchronizing clocks required for synchronous apparatuses is made possible by transferring transition point information of the synchronizing clock using a control information transfer area (overhead space of NNI standard) in a frame. When the distributed synchronizing clock frequency is set close to a frame repeat frequency, the number of the synchronizing clock transition point in one frame is either one of 0, 1 or 2. Therefore, it is possible to distribute the synchronizing clock by preparing a space where information for two transition points can be transmitted in the control information area. Moreover, overflow and underflow of synchronous information in nodes are prevented by having the information quantity applied in a LAN synchronize with the synchronizing clock and supplying the synchronizing clock to the synchronous apparatuses.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 is a block diagram showing an embodiment of an independent clocking LAN according to the present invention;

FIG. 2 is a diagram showing a utilization configuration of a multimedia LAN;

FIG. 3 is a frame construction diagram used in a LAN of the present invention;

FIG. 4 is a diagram showing a construction of a section overhead in a frame;

FIG. 5 is a block diagram showing a construction of a node constituting a LAN of the present invention;

FIG. 6 is a signal construction diagram of a clock pointer used in an embodiment of the present invention;

FIG. 7 is a pattern diagram of synchronizing clocks regenerated in an embodiment of the present invention;

FIG. 8 is a block diagram of a synchronizing clock generating circuit in an embodiment of the present invention;

FIG. 9 is a block diagram of a clock pointer generating circuit in an embodiment of the present invention;

FIG. 10 is a waveform diagram for explaining a jitter generating mechanism in case of synchronizing clock transfer;

FIG. 11 and FIG. 12 both show block diagrams of a stuffing portion in a master node in an embodiment of the present invention;

FIG. 13 is a distribution diagram of SOH spaces at stuffing buffer input/output;

FIG. 14 is a block diagram of a stuffing control portion in a general node; and

FIG. 15 is a frame format diagram of FDD-I which has been heretofore known.

DESCRIPTION OF THE PREFERRED EMBODIMENTS

Both FIG. 1 and FIG. 2 show constructions of multimedia LANs in an embodiment of an independent clocking LAN according to the present invention. FIG. 1 shows a part of FIG. 2 in detail for the convenience of explanation. In a multimedia LAN shown in FIG. 2, nodes 2-1 to 2-13 are connected in a ring form in a transmission line 1. The transmission line 1 is an optical fiber, and the transmission rate is at 155.52 Mb/s. NNI standards of CCITT are adopted for the transmission rate of a physical layer and a frame format. Namely, the transmission rate is at 155.52 Mb/s. A frame format of NNI standard will be explained later with reference to FIG. 3. It is possible to connect either one of a synchronous apparatus and an asynchronous apparatus to a multimedia LAN. In FIG. 2, Private Branch Exchanges (PBXs) 5-1 and 5-2, remote units (RSU) 6-1, 6-2 and 6-3 of the PBXs, picture units 7-1 and 7-2, multiplexers (MUX) 3-1, 3-2 and 3-3 for communicating with remote locations through high speed digital lines are connected as synchronous apparatuses. These apparatuses are connected with one another through channels which guarantee periodical information transfer. On the other hand, FDDI-I 4-1, 4-2, 4-3 and 4-4 which are LANs are connected directly to the transmission line 1 through nodes 2-3, 2-13, 2-4 and 2-12 of the multimedia LAN as asynchronous apparatuses. Work stations (WS), computers (HOST, CCP) and the like are connected to FDDI-I 4-1, 4-2, 4-3 and 4-4 directly or through further subordinate low speed LANs such as IEEE 802.3, 802.5 and the like. It is also possible to include them directly in the nodes. It is a matter of course to employ a multi-

plexer used in common to synchronous and asynchronous apparatuses. In a multimedia LAN, a plurality of synchronous and asynchronous apparatuses are connected with one another in a high speed, thus realizing information transmission and switching among apparatuses.

FIG. 3 shows a construction of a frame transmitted in the LAN. The frame is the same as a SONET frame which meets the NNI standard and is composed of 270 rows and 9 columns, and 2,430 bytes (270×9) are transmitted in about every 125 μs (depending on an oscillation frequency of a node). Thus, the transmission frequency is at approximately 155.52 MHz. Among 270 rows in the frame, the first 9 rows form a section overhead (hereinafter abbreviated as SOH) space, and are used for control and control information transfer among the nodes. The remaining space is used for information transfer. In order to make an information transfer rate in a multimedia LAN independently of the frequency of the node clock, information is transmitted by using a transmission block of 2,349 (261×9) bytes which is called a virtual container 4 (hereinafter abbreviated as VC-4). The positional relationship between VC-4 and the frame is not necessarily fixed, but the positional relationship between them is varied in accordance with the difference between the node clock and the rate of information which is transferred along the transmission line. Namely, it is possible to set a starting point of VC-4 at an arbitrary point (3 byte unit) of the information portion in the frame.

FIG. 4 shows the detail of a SOH space. For example, A1 and A2 show synchronous patterns, and a starting point of a frame is detected by periodical detection of A1 and A2 (the frame period is constant). Further, since frame synchronous patterns A1 and A2 appear periodically, it is able to prevent the occurrence of erroneous out-of-synchronization due to a bit error generated in transmission by deciding a synchronizing error (synchronization protection) with non-detection of the synchronous pattern in a plurality of times after frame synchronization is established. An AU pointer (including a stuffing space) in the fourth row of SOH shows the starting point of the above-mentioned VC-4. In the information transmission portion in the frame, addresses are added in 3 byte unit from the fourth row and the tenth column, and the address of the starting point of VC-4 is contained in the AU pointer. Accordingly, it is possible to find out the starting point of VC-4 by looking at the AU pointer. When the relative position between the frame and VC-4 is changed (called stuffing), the seventh column through the twelfth column of the fourth row are used. There are two types of stuffing, positive (positive stuffing) and negative (negative stuffing), which are used properly so as to compensate for the difference in rate in accordance with the relative magnitude between a frame repeat rate determined by the clock frequency of the node and an information transmission rate. When the frame repeat rate of the node is higher, it is required to shift the head of VC-4 in a direction in which the head address increases with respect to the frame. Therefore, adjustment is made by blanking the fourth row and the tenth to twelfth columns (positive stuffing). Conversely, when the frame repeat frequency is lower than the information transmission rate, the head position of VC-4 is altered by transferring the information by using the fourth row and the seventh to ninth columns, too (negative stuffing). Namely, the difference between the frame repeat rate and the information transmission rate is adjusted by varying the size of the information transfer space in one frame. Generation of stuffing is informed to a downstream node by altering the pointer value. The deviation between the frame repeat rate and the information transmission rate is specified by an allowable stuffing frequency. Since stuffing can be performed only once in four frames according to the NNI standard, the deviation of three bytes is allowable for four frames. With this, it is required that the node clock frequency deviation of each node falls within ±309 ppm (=3/(2,430×4)) with respect to the information transmission rate. Further, in FIG. 4, D1-D12 show data spaces in 12 bytes called data communication channels and can be used for control information transmission among the nodes. In the present embodiment, the transition point information of the synchronizing clock is transmitted by utilizing these spaces. Bytes B1, B2, C1, E1, E2, F1, K1, K2, Z1 and Z2 are not required for explaining the present invention. Hence, the explanation thereof is omitted herein.

Returning now to FIG. 1, information transfer and clock distribution in a multimedia LAN will be described. In FIG. 1, only the nodes 2-8 to 2-11 in FIG. 2 are shown, and other nodes are omitted. Interface portions with apparatuses connected to nodes are shown with respect to the node 2-9 only. The node 2-8 is the master node, and supplies a common synchronizing clock (8 KHz) to other nodes. In FIG. 1, a path 12 shows a distribution path of a synchronizing clock, and a path 16 shows a transmission path for information. From a physical point of view, two paths are multiplexed, and transmission is performed using a single transmission path. In the multimedia LAN, an external clock 9 (8 KHz) supplied from the outside is distributed among all the nodes 2-8 . . . 2-11 as a common synchronizing clock and is supplied to synchronous terminals connected to the nodes from respective nodes. Further, the information transfer rate is determined by outputting VC-4 synchronously with the external clock 9. Respective nodes 2-9 . . . 2-11 transfer a common synchronizing clock information from the master node 2-8 to following nodes by means of relay units or repeaters 11-9 to 11-11. In each node, clock jitter generated at the time of clock transfer is reduced and a synchronizing clock is supplied to a synchronous apparatus 7-2 by means of a phase lock loop 13-9 (omitted with respect to other nodes). Further, respective nodes include oscillators 15-8 to 15-11 of the own station, determine frame periods with oscillated node clocks and send them to following nodes. Stuffing buffers 14-8 to 14-11 are used for absorption of the difference between a receiving frequency and a transmitting frequency. In case the information quantity stored in the buffer is varied by ±3 bytes and more from a central value of a buffer capacity, stuffing is executed and adjustment is made so that the information quantity applied to the ring becomes constant in the whole LAN. The synchronizing clock information arrives at the node almost periodically, but does not synchronize completely with the distributed synchronizing clock for a short time by the influence of stuffing. Therefore, this variation portion is controlled in a delay control circuit 16-9. In the master node 2-8, it is required to provide a function of controlling the delay so that the delay which makes a round in the ring becomes integer times as long as the frame period so that no information deficiency is produced in transferring operation. Usually, a buffer having a capac-

ity of approximately one frame portion is prepared for that purpose separately from the stuffing buffer (for example, see JP-B-61-44426). This function is described along with a stuffing buffer **14-8** with reference to FIG. **1** for the sake of simplicity.

FIG. **5** shows the construction of the node **2-9** in FIG. **1** in detail. Same numbers are affixed to those parts that are the same as components shown in FIG. **1**. The node **2-9** is divided into three regions (A), (B) and (C) with one-dot chain lines depending on the type of used clock. The region (A) is operated by means of a received clock regenerated by an optical receiver **21** from received data. The region (B) is operated by means of a self-node clock supplied from a clock source **15-9** included in the node. The region (C) is operated by means of a synchronizing clock supplied from the master node. The transmission frequency is at 155.52 MHz, but the information is processed in one byte unit in the node. Therefore, the inside of the node is operated by a clock at the transmission frequency of $\frac{1}{8}$ of 155.52 MHz, viz., 19.44 Mhz except a frame synchronous circuit **22**. Accordingly, the oscillation frequency of the clock source **15-9** of the node is at 19.44 Mhz.

In the region (A), the received optical signal is converted into an electrical signal in the optical receiver **21**, and a transmission clock (155.52 MHz), viz., a clock in an upstream node is extracted as the received clock. The extracted received clock is supplied to a frame synchronizing circuit **22**, a multiplexing and demultiplexing/SOH extraction circuit **23** and a clock generation circuit **26** in the region (A). A starting point of frame is detected by the frame synchronization circuit **22**. The multiplexing and demultiplexing/SOH extraction circuit **23** applies serial-parallel conversion to a signal at 155.52 Mb/s so as to convert into a byte unit of 19.44 MHz and also extracts the SOH portion of the received frame. An elastic buffer **24** in the region (B) is used for the purpose of absorbing a phase difference and a frequency difference between the received (transmission) clock and node clock. Among the received frame information, the portion of VC-4 is written in the elastic buffer **24** in byte unit. The information for showing the head of VC-4 is written at the same time. An access control circuit **25** monitors an empty status of the elastic buffer **24**, and causes the VC-4 data to synchronize with the node clock when VC-4 data are in existence by reading the data using the node clock in byte unit. Since the information quantity of VC-4 is **261/270** of the information quantity in the frame (see FIG. **3**), overflow of the elastic buffer **24** is not generated if the node clock frequency deviation is controlled within 3.8% (the frequency deviation is set within 308 ppm practically in order to prevent overflow of the stuffing buffer from occurring). Further, an access control circuit **25** performs relaying of information and information switching to and from synchronous and asynchronous apparatuses connected to the nodes. As a method of transferring synchronous and asynchronous information, a time division type system in which the inside of VC-4 is divided into two regions, one for transferring synchronous information and another for transferring asynchronous information, that is, VC-4 is divided into regions called slots and each slot is divided for synchronous information and asynchronous information (slotted ring) and so forth are adopted. Further, various techniques (see "The Data Ring Main: An Introduction to Local Area Network", written by David C. Flint, for instance) are well known as an access system to a ring of asynchronous information, both systems are applicable. The present invention relates principally to a construction of a physical layer. Hence, explanation of an access system to a ring is omitted herein. The synchronous apparatus is connected to the access control circuit **25** through a synchronous apparatus interface **30**, a synchronizing buffer **28** for receiving data and a synchronizing buffer **29** for transmitting data (corresponding collectively to **16-9** shown in FIG. **5**). The synchronous apparatus interface **30** terminates a protocol of the synchronous apparatus and converts it into an information format in the ring. Further, synchronizing buffers **28** and **29** for receiving and transmitting data are used for absorbing phase difference and instantaneous frequency difference between a common synchronizing clock and a node clock. The synchronous apparatus interface **30** belongs to the region (C) and is operated with the common synchronizing clock. On the other hand, the asynchronous apparatus is connected through an asynchronous apparatus interface **31**, but it is not required to guarantee periodicity of information transfer. Therefore, the asynchronous interface **31** is operated with a node clock in a similar manner as the access control unit **25**.

Information is sent out from the node **2-9** in a specified frame period with the node clock. Therefore, the information quantity which VC-4 can transfer is different in general from the information quantity of VC-4 which has been input to the node **2-9**. A stuffing function is used in order to absorb the difference in the information quantity. The output of the access control circuit **25** is written into a stuffing buffer **14-9** in byte unit. A stuffing control and frame generating circuit **33** generates a frame which has been explained with reference to FIG. **3** and FIG. **4** and reads the information in byte unit from the stuffing buffer **14-9** at the time of outgoing of VC-4 and sends out the information. With the stuffing operation, the information quantity in the stuffing buffer **14-9** is monitored, and execution is performed when a predetermined threshold is exceeded. For example, when the threshold is set to $\frac{1}{4}$ of the stuffing buffer capacity ± 3 bytes, negative stuffing is performed when the capacity of information accumulated in the stuffing buffer exceeds $\frac{1}{4}$ of the stuffing buffer capacity $+3$ bytes, and the information quantity which can be transferred in one frame is increased for adjustment. Further, when the capacity of information accumulated in the stuffing buffer is lower than $\frac{1}{4}$ of the stuffing buffer capacity -3 bytes, positive stuffing is performed, and the information quantity which can be transferred in one frame is reduced for adjustment. As described previously, it is required to control the frequency deviation of the node clock to fall within ± 308 ppm in order not to generate overflow and underflow of the stuffing buffer **14-9**. In a SOH insertion and multiplex (MUX) circuit **34**, SOH information is inserted and information in byte unit is multiplexed so as to obtain information at 155.52 Mb/s. The node clock (19.44 MHz) is multiplied by light by using a PLL and the like so as to become a clock at 155.52 MHz, which is supplied to a multiplexing circuit **34** and an optical transmitter **35** (connection omitted in the figure). The information applied with series conversion is converted into an optical signal in an optical transmitter **35** and sent out to an optical fiber which is a transmission line.

Further, the synchronizing clock information existing in the SOH which is extracted in the SOH extraction circuit **23** is sent to the clock generation circuit **26**

through a signal line 41, and a synchronizing clock is generated. The generated synchronizing clock is supplied to the synchronizing buffer 28 for receiving data and the synchronous apparatus interface 30 after jitter is suppressed in a PLL 13-9. Further, the generated synchronizing clock is synchronized with the node clock in a synchronization circuit 27 (details will be explained with reference to an embodiment shown in FIG. 8). In a clock pointer generation circuit 32, transition points of the synchronizing clock synchronized with the node clock are counted from the starting point of frame, and the counted value is added to the SOH insertion circuit 34 as transition point information of the synchronizing clock. The starting point of frame is known with a frame head signal 36 from the stuffing control/frame generation circuit 33. In the SOH insertion circuit 34, the transition point information of the synchronizing clock is inserted into a data communication channel of SOH, and is sent out to the following node as synchronizing clock information.

The distribution method of the synchronizing clock will be described in detail hereinafter. FIG. 6 shows a transfer format of transition point information of a synchronizing clock. A region in 5 bytes in the whole is used. For example, transfer is made using D1–D5 of the data communication channels of the SONET frame explained with reference to FIG. 4. Taking a case where there are two transition points of the clock in one frame into consideration, two clock pointers showing the transition point of the synchronizing clock are transferred. The last one byte is a CRC (cyclic redundancy check) code for error check. FIG. 7 shows the relationship between the transfer format and the regenerated clock. The synchronizing clock (8 KHz) is sampled with the node clock ($19.44 \pm \beta$MHz) of a previous node, and the transition point information is transferred in the format shown in FIG. 6. Since the period ($125\ \mu s \pm \alpha$) of the common synchronizing clock and the frame period generated by a previous node are different from each other, there are a case in which the transition point is nonexistent (case (III) shown in FIG. 7), a case in which there is one transition point (case (I)) and a case in which there are two transition points (case (II)) in one frame. In each node, the frequency of a transmission clock (155.52 MHz) regenerated from a received optical signal is divided by eight so as to regenerate a node clock of a previous node, and a common sycnhronizing clock is regenerated using the node clock and the transition point information of the synchronizing clock. 2,430 pcs. of node clocks are included in one frame, which are counted from the starting point of frame, and the pointer (FIG. 6) shows the clock in which the synchronizing clock has varied. Thus, it is possible to express the pointers (A) and (B) in 12 bits, but 2 bytes are used for each pointer in order to delimit information in byte unit. The pointer (A) shows the first synchronizing clock transition point in the frame, and the pointer (B) shows the second transition point. When there is not relevant transition point, all bits are set to "1". In the case (I) of FIG. 7, there is one transition point in the frame, which is shown with the pointer (A). In the receiving node, the regenerated node clocks are counted from the starting point of the received frame, and a common synchronizing clock is regenerated by generating a pulse when N1 (value of pointer (A)) pcs. are counted. In the case (II), there are two transition points, and pulses are generated each time when N1 and N2 clocks are counted. The case (III) shows a case in which there is no transition point in the frame. All bits are set to "1" in both pointers (A) and (B). When a CRC error is generated in the pointer, received pointer information is abolished, and a transition point is determined by counting 2,430 clocks from the last transition point of the regenerated synchronizing clock. Since it is usually possible to control the deviation between the node clock and the synchronizing clock small, it is possible with the above to control the influence by a transmission error small even if the transmission error is generated.

FIG. 8 shows details of the clock generation circuit 26 and the synchronization circuit 27 of FIG. 5. For the purpose of simplifying explanation, a check processing circuit of CRC errors is omitted. The frequency of a transmission clock regenerated in the optical receiver 21 is divided by eight by the multiplexing and demultiplexing/SOH extraction circuit 23, and sent to the clock extraction circuit 26 as a regenerated clock ($19.44 \pm \beta$ MHz) through the signal line 41 together with extracted clock pointers (A) and (B). The received two pointers (A) and (B) are used for generating transition point information of a synchronizing clock in a next frame. Accordingly, subordinate twelve bits of the pointers (A) and (B) are loaded on latches 45 and 50 by means of a frame starting signal 36. On the other hand, two twelve bit counters 43 and 48 are reset by means of the frame starting signal 36 and start counting up. When the value of the counter 43 (or 48) and the value of the latch 45 (or 50) are in agreement with each other, an output of a comparator 44 (or 49) reaches "H", and sets a set/reset type flip-flop 47 (or 52). The flip-flops 47 and 52 are reset by signals which are delayed in delay elements 46 and 51, respectively. Therefore, when the counter value agrees with the pointer value, a pulse is generated at that point. Since the maximal counter value is 2,430, the pointer and the counter are not in agreement with each other when all the bits of the pointer are at "1", and no clock is generated. Since the output of the flip-flop 47 shows the transition point by the pointer (A), and the output of the flip-flop 52 shows the transition point by the pointer (B), it is possible to regenerate a synchronizing clock by obtaining an OR of two outputs with an OR gate 53. Since the output of the clock generation circuit 26 is in synchronism with the regenerated node clock of a previous node, it is impossible to use the output as it is in a clock pointer generation circuit 32 which is operated with a self-node clock. Therefore, the output is synchronized with the self-node clock in a clock synchronization circuit 27. The synchronization circuit 27 is composed of cascase connection in two stages of two edge-trigger flip-flops 54 and 55. The self-node clock is supplied to two flip-flops 54 and 55. Since the input of the flip-flop 54 and the self-node clock are asynchronous with each other, the output becomes unstable sometimes. However, the output is synchronized with the self-node clock by taking the output of the flip-flop 54 into the flip-flop 55 when the unstable state is dissolved. An output 38 of the clock synchronization circuit 27 is added to the clock pointer generation circuit 32.

FIG. 9 shows the detail of the clock pointer generation circuit 32. The number of self-node clocks from the starting point of frame in a self-node to the transition point of the synchronized synchronizing clock 38 is counted by a 12-bit counter 64. A 2-bit counter 63 counts the number of synchronizing clock transition points in one frame. Both counters 64 and 63 are reset

by a frame starting signal 36. Further, all the bits of latches 62 and 66 are set to "1" by the frame starting signal 36. With this, "1" is output for all in case there is no clock transition point in the frame. The bit b0 output of the counter 63 shows that the transition point of the clock is either the first (b0=1) or the second (b0=0) in the frame. The output of an AND gate 61 is varied at the first transition point, and the value of the counter 64 at that time is taken into the latch 62. Further, the output of an AND circuit 65 is varied at the second transition point and the counter value at the transition point is taken into the latch 66. The output of the counter 63 is varied at the transition point of the synchronizing clock, and a delay circuit 71 is inserted in order to prevent the pulse width of the outputs of the AND gates 61 and 65 from narrowing. The number of synchronizing clock transition points and the positions of the transition points in one frame are found when looking at outputs 69 and 70 of the counter 63 and outputs 67 and 68 of the latches 62 and 66. Thus, a clock pointer, viz., transition point information of the synchronizing clock which is sent to a node on the next stage is generated using the above findings.

Next, clock jitter in the above-mentioned synchronizing clock distribution will be described. The synchronizing clock is repeated in succession in respective nodes, but jitter is generated when the synchronization circuit 27 of each node performs synchronization. FIG. 10 shows a generating mechanism of the jitter. As it is understood from the figure, the transition point of the generated synchronizing clock (input of F/F (54)) and the transition point of the synchronizing clock after synchronization (output of F/F (55)) are shifted from each other by one clock period + Δx. Since node clock frequencies of respective nodes are different from one another, Δx is varied time-wise so as to form a jitter. Since Δx is varied from 0 to 50 ns (1/19.44 MHz) at the maximum, the maximum value of the jitter reaches 50 ns×number of nodes in the worst case. However, this jitter is suppressed by the PLL. Since the jitter attenuation quantity of the PLL is in proportion to the jitter frequency in general, it is possible to attenuate high frequency jitter to such a level that the jitter causes no problem in the PLL. Accordingly, the jitter in a low frequency becomes an issue. In order to evaluate the jitter quantity in case the number of connected nodes is increased, a case of the number of nodes at 128 is evaluated for instance. Now, a case in which the jitters of 50 ns at the maximum in respective nodes are added to become jitter of a single frequency at the 128th node is considered as the worst case. The maximum amplitude of the jitter becomes 25 ns×127 transfers=3.3 μs. Since the jitter is usually specified at 10 Hz and above (the jitter at less than 10 Hz is called "wander"), the jitter at 10 Hz is considered. A case in which the jitter added in the worst case shows a sinusoidal wave at 10 Hz is considered. In this case, all the jitter power is concentrated to 10 Hz. When it is assumed that the jitter attenuation quantity at 10 Hz in the PLL is 30 dB (a value which is able to be realized easily with existing techniques by using a voltage controlled crystal oscillator), the jitter of the PLL output becomes approximately 100 ns which is an allowable value (for example, a user/network interface at 1.5 Mb/s is specified in TTC standard JT-1431, but 3.2 μs is specified in the frequency range from 10 Hz to 120 Hz as a jitter quantity to be allowed by the terminal). It is also possible to control the jitter quantity by altering the jitter attenuation quantity by varying parameters of the PLL and by varying the frequency of sampling the synchronizing clock. Since sampling is made at 19.44 MHz in the embodiment, the jitter generated during relaying operation was 50 ns at the maximum, but it is possible to reduce the jitter quantity by increasing the sampling frequency because the generated jitter is reduced in inverse proportion to the sampling frequency.

In the next place, stuffing for absorbing the difference between the node clock frequency and the information transfer rate in each node will be described in detail. In the embodiment shown in FIG. 1, the information transfer rate is specified by an external clock 9. Namely, when synchronization between the starting point of VC-4 and the external clock 9 is off, stuffing is performed to have both of them in agreement with each other in order to cause the starting position of VC-4 generated by the master node 2-8 to synchronize with the external clock 9. On the other hand, stuffing is performed in general nodes 2-9 to 2-11 except the master node 2-8 in order to send out the information quantity which is sent from a previous node without excess and deficiency with the self-node clock. Accordingly, stuffing algorithms are different between the master node 2-8 and general nodes 2-9 to 2-11.

FIG. 11 shows a composition for realizing stuffing in the master node 2-8. Namely, a construction 33' of a circuit corresponding to a stuffing control/frame generation circuit 33 of the general node 2-9 is shown. Stuffing in the master node is executed so that the external clock and the starting point of VC-4 generated by the master node are compared with each other and both phases fall within a fixed value. With this, it is possible to make the rate of information outputted from the master node agree with the external clock. In FIG. 11, a clock input 17 is an output obtained by applying the external clock 9 of FIG. 1 to the PLL 10 and reducing the jitter. This clock input 17 is synchronized with a node clock from a clock source 15-8 in a synchronization circuit 82 (the construction thereof is the same as 27 shown in FIG. 8), and is compared with a starting point signal 87 of VC-4 which is generated by a frame generation control circuit 86. A counter 83 is reset by the synchronized external clock and counted up with the node clock. Therefore, the number of node clocks from the startup of the synchronized external clock is counted. It is possible to know the phase difference between the external clock and the starting point of VC-4 by loading the value of the counter 83 onto a latch 84 by a VC-4 starting point signal 87 which is outputted from the frame generation control circuit 86. Since the phase difference is distributed from 0 to 2,429 (number of node clocks in one frame - 1), it is possible to control so that the starting point of VC-4 falls within three clocks from the synchronized external clock by performing negative stuffing when the phase difference is, for example, a value from 4 to 1,215 and positive stuffing is a value from 1,215 to 2,426. Decision of the phase difference is made by means of a decision circuit 85, and the result is sent to the frame generation control circuit 86, thus executing stuffing.

FIG. 12 shows a construction of another embodiment of a stuffing control portion in the master node. In the present embodiment, it is controlled so that the difference between the number of bytes of VC-4 which are actually sent out during one period of the synchronized external clock and the number of bytes (261×9=2,349) which are to be sent originally becomes a fixed value

and less. In FIG. 12, those parts that are the same as FIG. 11 are affixed with same numbers, and explanation thereof is omitted. Since the counter 83 is reset by a signal obtained by synchronizing the external clock 17, it is possible to count the VC-4 output signal supplied by the frame generation control circuit 86 for one period portion of a signal obtained by synchronizing the external clock. At the time when counting is terminated, the difference between 23 and 49 is obtained by means of a subtraction circuit 89 and accumulated with an accumulator 88. It is decided by the decision circuit 85 that the accumulated value exceeds +3 bytes, and the result is sent to the frame generation circuit 86, thereby to control stuffing in the master node.

In case the external clock 9 cannot be utilized, the frequency of the clock output from the clock source 15-8 of the master node 2-8 is demultiplied to generate a signal of 8 KHz, which is used as a synchronizing clock source. In this case, stuffing is not generated because the node clock and the synchronizing clock are in synchronism with each other in the master node.

Next, a stuffing control portion in a general node will be explained in detail. FIG. 13 shows information which is inputted and outputted in and from the stuffing buffer 14-9 shown in FIG. 5. Hatched portions in the figure show SOH spaces. The frame structure is shown in FIG. 3, but the frame is transmitted from left to right and from top to bottom successively. Therefore, the SOH space including 9 bytes appears periodically as shown in FIG. 13. Further, since frame periods and starting points of frame of reception and transmission are independent, respectively, the SOH spaces of transmission and reception are not in synchronism with each other as shown in FIG. 13, and the phase difference is varied time-wise. Accordingly, such a problem arises that when the node determines whether or not stuffing is to be performed based on an information quantity in the stuffing buffer 14-9. For example, since the input is the SOH space in 9 bytes during the period from a point a to a point b in FIG. 13, information is not written in the stuffing buffer, but the output is read out of the stuffing buffer 14-9 because the output is an information space. Thus, the information quantity in the stuffing buffer at the point b is reduced by 9 bytes as compared with the point a. In such a manner, the information quantity in the stuffing buffer 14-9 depends on the time of observation and is varied by ±9 bytes. In order to avoid such a problem, a system is adopted, in which the position in the buffer where information has been read is stored when information in one frame is read out of the stuffing buffer 14-9 in byte unit and it is decided whether stuffing is executed or not by a mean value of one frame portion of the information quantity in the stuffing buffer 14-9. According to this system, stuffing is performed correctly because reduction of the information quantity in the stuffing buffer due to writing in the SOH space and increase of the information quantity in the stuffing buffer due to writing of SOH are offset each other by averaging even under a status such as shown in FIG. 13.

FIG. 14 shows a construction of an embodiment of the stuffing buffer 14-9 and the stuffing control and frame generation circuit 33 shown in FIG. 5 for explaining the above-mentioned algorithm. The stuffing buffer 14-9 is composed of a buffer memory 93 which stores information in byte unit and counters 94 and 95 which control write and read addresses, respectively. The counter 94 is counted up by a write signal 92 every time information is written from an access control circuit 25 through a line 39, and the counter 95 is counted by a read signal 104 every time information is read by the stuffing control and frame generation circuit 33. The counter value is reset when it reaches the maximum capacity of the buffer 93. Thus, it is possible to know the information quantity in the buffer 93 by obtaining the difference between both counters 94 and 95 by a subtraction circuit 96. The results thereof are accumulated for one frame portion using an adder 97 and a latch 98. In order to obtain accumulation for one frame portion, the latch 98 is reset by a frame starting signal 101, and supplies a clock 105 only when information (excluding the SOH space) is read out. A decision circuit 99 decides whether stuffing is to be performed in a next frame from the accumulated value of the information quantity in the buffer 93 and the number of transfer bytes in one frame at the point when accumulation of one frame portion is completed. Namely, the accumulated value determines stuffing or no stuffing with the number of VC-4 transfer bytes x (the maximum capacity of the buffer 93/2±3 bytes) as a boundary. The number of transfer bytes in one frame has only three types, that is, 2,346 bytes (positive stuffing), 2,349 bytes (no stuffing) and 2,352 bytes (negative stuffing) depending on the existence of stuffing in that frame, and is informed to the decision circuit 99 from a frame generation control circuit 102 through a signal line 106. The result of decision is transferred to a frame generation control circuit 102 through a signal line 100, thus executing stuffing.

An embodiment of the present invention has been described above, but it is clear that the present invention is not limited to the above-mentioned embodiment. Explanation has been made so far with respect to a single frame. A plurality of frames are applied practically with time division multiplexing for transmission in many cases, but a case of transmission applied with time division multiplexing is included in the present invention as a matter of course.

For example, when four frames are transmitted with time division multiplexing being applied thereto (information transmission rate is 155.52×4 Mbps), it may be arranged so as to perform stuffing for four frames at the same time instead of performing stuffing for each frame (155.52 Mbps).

According to the present invention, it is possible to compose a multimedia LAN which has no jitter accumulation to a transmission clock and is able to control the jitter of the synchronizing clock to such a level that has no problem. Furthermore, it is possible to use a fixed length frame which is standardized internationally, and to realize a LAN which is durable against transmission errors and in which the physical transmission rate and the logical transmission are equal to each other.

We claim:

1. An independent clocking local area network in which a plurality of nodes are connected through transmission lines, wherein each of said nodes comprises:

means for extracting a received clock signal;

a clock source which generates an independent node clock signal;

frame generating means for generating a fixed length frame with an oscillation frequency of said independent node clock signal as a reference;

means for regenerating a synchronizing clock signal which has been sent out by an upstream node using transition point information provided in a specific space of a received frame;

synchronization means for synchronizing the regenerated synchronizing clock signal with the independent node clock signal; and

means for detecting a transition point of the synchronizing clock signal synchronized by said synchronization means and setting said detected transition point in a specific space of the fixed length frame generated by said frame generation means as the transition point information of the synchronized synchronizing clock signal.

2. An independent clocking local area network according to claim 1, wherein said means for detecting and setting the transition point in each of said nodes further comprises:

means for counting the time from a position of said generated fixed length frame to the transition point of the synchronizing clock signal synchronized with said independent node clock signal; and

means for inserting a value obtained as a result of counting into said specific space of said generated fixed length frame.

3. An independent clocking local area network according to claim 1, wherein said generated fixed length frame is composed of a SONET frame, and the specific space in said generated fixed length frame which transfers the transition point information of said synchronizing clock signal is a data communication channel.

4. A local area network according to claim 1, wherein at least one node of said plurality of nodes is a master node for feeding said synchronizing clock signal as a master synchronizing clock signal to remaining nodes, and said master node comprises means for using an external clock signal or said independent node clock signal as said master synchronizing clock signal, and setting a transition point of said master synchronizing clock signal in a specific space of said generated fixed length frame and means for adjusting an information quantity carried in said generated fixed length frame so that an information quantity specified by said master synchronizing clock signal is transferred to the next node.

5. A node for a local area network, comprising:

receiving means for receiving information of a fixed length frame;

means for extracting a received clock signal;

a clock source which generates an independent node clock signal;

frame generation means for generating a fixed length frame with an oscillation frequency of said independent node clock signal as a reference;

regenerating means for reading transition point information of a synchronizing clock signal out of a specific space of said fixed length frame received by said receiving means and for regenerating said synchronizing clock signal which has been sent from an upstream node using said received clock signal;

synchronization means for synchronizing said regenerated synchronizing clock signal with said independent node clock signal generated by said clock source;

means for detecting a transition point of said synchronized regenerated synchronizing clock signal in said received frame; and

means for inserting the transition point of said synchronized regenerated synchronizing clock signal into a specific space in the generated fixed length frame.

6. An independent clocking local area network composed of a plurality of nodes and transmission lines connecting said plurality of nodes, wherein each of said nodes comprises:

means for extracting a received clock signal;

a clock source which generates an independent node clock signal;

storage means for storing received information temporarily;

means for generating a fixed length frame with an oscillation frequency of said independent node clock signal as a reference; and

information outgoing quantity control means for increasing information quantity which is carried by each frame when information quantity stored in said storage means becomes more than a predetermined first reference value, and decreasing information quantity which is carried by each frame when information quantity stored in said storage means becomes less than a predetermined second reference value;

wherein each of said nodes comprise means for transferring a synchronizing clock signal to be fed to synchronous terminals by setting a transition point of said synchronizing clock signal in a specific space of said generated fixed length frame.

7. An independent clocking local area network according to claim 6, wherein said transfer means includes means for counting the time from a position of said generated fixed length frame to a transition point of said synchronizing clock signal, and means for inserting a value of the counting result into said specific space.

8. An independent clocking local area network according to claim 6, wherein said plurality of nodes are connected in a ring form.

9. An independent clocking local area network according to claim 6, wherein said generated fixed length frame is composed of a SONET frame, and said specific space in said generated fixed length frame which transfers the transition point of said synchronizing clock signal is a data communication channel.

10. A local area network according to claim 6, wherein at least one node of said plurality of nodes is a master node for feeding said synchronizing clock signal as a master synchronizing clock signal to remaining nodes, and said master node comprises means for using an external clock signal or said independent node clock signal as the master synchronizing clock signal and setting a transition point of said master synchronizing clock signal in said specific space of said generated fixed length frame and means for adjusting an information quantity carried in said generated fixed length frame so that an information quantity specified by said master synchronizing clock signal is transferred to the next node.

11. An independent clocking local area network composed of a plurality of nodes and transmission lines connecting said plurality of nodes, wherein each of said nodes comprises:

means for extracting a received clock signal;

a clock source which generates an independent node clock signal;

storage means for storing received information temporarily;

means for generating a fixed length frame with an oscillation frequency of said independent node clock signal as a reference; and

information outgoing quantity control means for increasing information quantity which is carried by each frame when information quantity stored in said storage means becomes more than a predetermined first reference value, and decreasing information quantity which is carried by each frame when information quantity stored in said storage means becomes less than a predetermined second reference value;

wherein said information outgoing quantity control means includes a means for increasing or decreasing the information quantity in each single frame in accordance with a result of accumulating information indicating the quantity of information stored in said storage means over the duration of at least one frame or a result of averaging information indicating the quantity of information stored in said storage means over the duration of at least one frame;

wherein each of said nodes comprise means for transferring a synchronizing clock signal to be fed to synchronous terminals by setting a transition point of said synchronizing clock signal in a specific space of said generated fixed length frame.

12. An independent clocking local area network according to claim 11, wherein said generated fixed length frame is composed of a SONET frame, and said specific space in said generated fixed length frame which transfers the transition point of said synchronizing clock signal is a data communication channel.

13. A local area network according to claim 11, wherein at least one node of said plurality of nodes is a master node for feeding said synchronizing clock signal as a master synchronizing clock signal to remaining nodes, and said master node comprises means for using an external clock signal or said independent node clock signal as the master synchronizing clock signal and setting a transition point of said master synchronizing clock signal in said specific space of said generated fixed length frame and means for adjusting an information quantity carried in said generated fixed length frame so that an information quantity specified by said master synchronizing clock signal is transferred to the next node.

14. An independent clocking local area network according to claim 11, wherein said plurality of nodes are connected in a ring form.

15. A local area network according to claim 8, wherein at least one node of said plurality of nodes is a master node for feeding said synchronizing clock signal as a master synchronizing clock signal to remaining nodes, and said master node comprises means for using an external clock signal or said independent node clock signal as the master synchronizing clock signal and setting a transition point of said master synchronizing clock signal in said specific space of said generated fixed length frame and means for adjusting an information quantity carried in said generated fixed length frame so that an information quantity specified by said master synchronizing clock signal is transferred to the next node.

16. An independent clocking local area network according to claim 11, wherein said generated fixed length frame is composed of a SONET frame, and said specific space in said generated fixed length frame which transfers the transition point of said synchronizing clock signal is a data communication channel.

17. A local area network according to claim 9, wherein at least one node of said plurality of nodes is a master node for feeding said synchronizing clock signal as a master synchronizing clock signal to remaining nodes, and said master node comprises means for using an external clock signal or said independent node clock signal as the master synchronizing clock signal and setting a transition point of said master synchronizing clock signal in said specific space of said generated fixed length frame and means for adjusting an information quantity carried in said generated fixed length frame so that an information quantity specified by said master synchronizing clock signal is transferred to the next node.

18. A local area network according to claim 6, wherein at least one node of said plurality of nodes is a master node for feeding said synchronizing clock signal as a master synchronizing clock signal to remaining nodes, and said master node comprises means for using an external clock signal or said independent node clock signal as the master synchronizing clock signal and setting a transition point of said master synchronizing clock signal in said specific space of said generated fixed length frame and means for adjusting an information quantity carried in said generated fixed length frame so that an information quantity specified by said master synchronizing clock signal is transferred to the next node.

* * * * *

US005602841A

United States Patent [19]

Lebizay et al.

[11] Patent Number: 5,602,841

[45] Date of Patent: Feb. 11, 1997

[54] **EFFICIENT POINT-TO-POINT AND MULTI-POINT ROUTING MECHANISM FOR PROGRAMMABLE PACKET SWITCHING NODES IN HIGH SPEED DATA TRANSMISSION NETWORKS**

[75] Inventors: **Gerald Lebizay**, Vence; **Jean M. Munier**, Cagnes sur Mer; **Andre Pauporte**, La Colle sur Loup; **Victor Spagnol**, Cagnes sur Mer, all of France

[73] Assignee: **International Business Machines Corporation**, Armonk, N.Y.

[21] Appl. No.: **404,800**

[22] Filed: **Mar. 15, 1995**

[30] **Foreign Application Priority Data**

Apr. 14, 1994 [EP] European Pat. Off. 94480030

[51] **Int. Cl.**[6] **H04L 12/56**

[52] **U.S. Cl.** **370/413**

[58] **Field of Search** 370/58.2, 60, 60.1, 370/61, 79, 94.1, 94.2

[56] **References Cited**

U.S. PATENT DOCUMENTS

5,365,519	11/1994	Kozaki et al.	370/60
5,393,280	2/1995	Zheng	370/60

FOREIGN PATENT DOCUMENTS

0575281	12/1993	European Pat. Off.	H04L 12/18
0579567	1/1994	European Pat. Off.	H04L 12/56
0632625	1/1995	European Pat. Off.	H04L 29/06
9102420	2/1991	WIPO	H04L 12/56

OTHER PUBLICATIONS

Globecom '91, vol. 1, 12-92 NY pp. 104-110.
Int. Journal of Digital and Analog Communication Systems, vol. 5, 1992 pp. 29-37.
Proceedings of the IEEE '92 Custom Integrated Circuits Conf., 5-29, NY, pp. 14.4.1-14.4.4.

Primary Examiner—Douglas W. Olms
Assistant Examiner—Russell W. Blum
Attorney, Agent, or Firm—Stephen T. Keohane; John J. Timar

[57] **ABSTRACT**

The present invention relates to an efficient point-to-point and multi-points routing system and method for programmable data communication adapters in packet switching nodes of high speed networks. The general principles of this efficiency are the following:

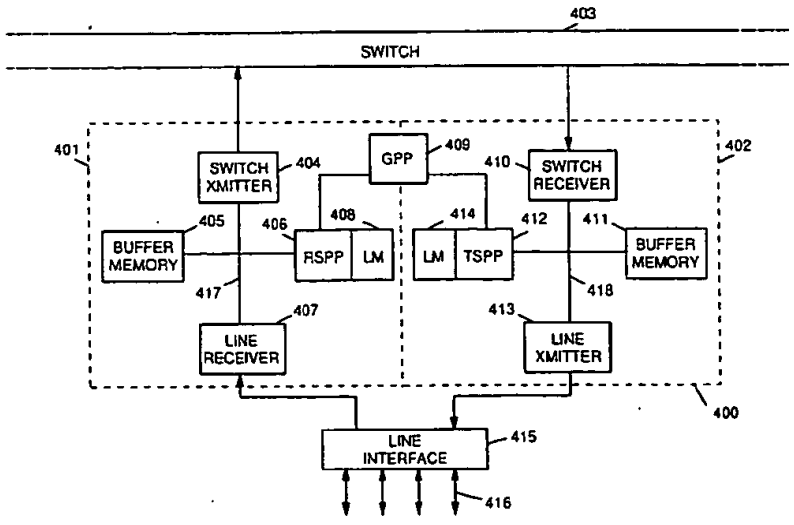
First, data packets are never copied, only packet pointers are copied for each destination: Space in Buffer Memory is saved, the number of instructions is significantly reduced improving the packet throughput (number of packets per seconds that the adapter is able to transmit). and the routing is independant of the packets length.

Second, no overhead is generated by the multi-points mechanism in the real time procedures: the underrun/overrun problems on the ouputs are reduced and the efficiency of the adapter in term data throughput (bits per second) is significantly improved.

Third, each output is processed independently by means of interrupts: lines are managed in real time and lines of different speed or protocol can be supported in parallel.

Fourth, the release of the resources is entirely realized on a non priority mode.

19 Claims, 19 Drawing Sheets



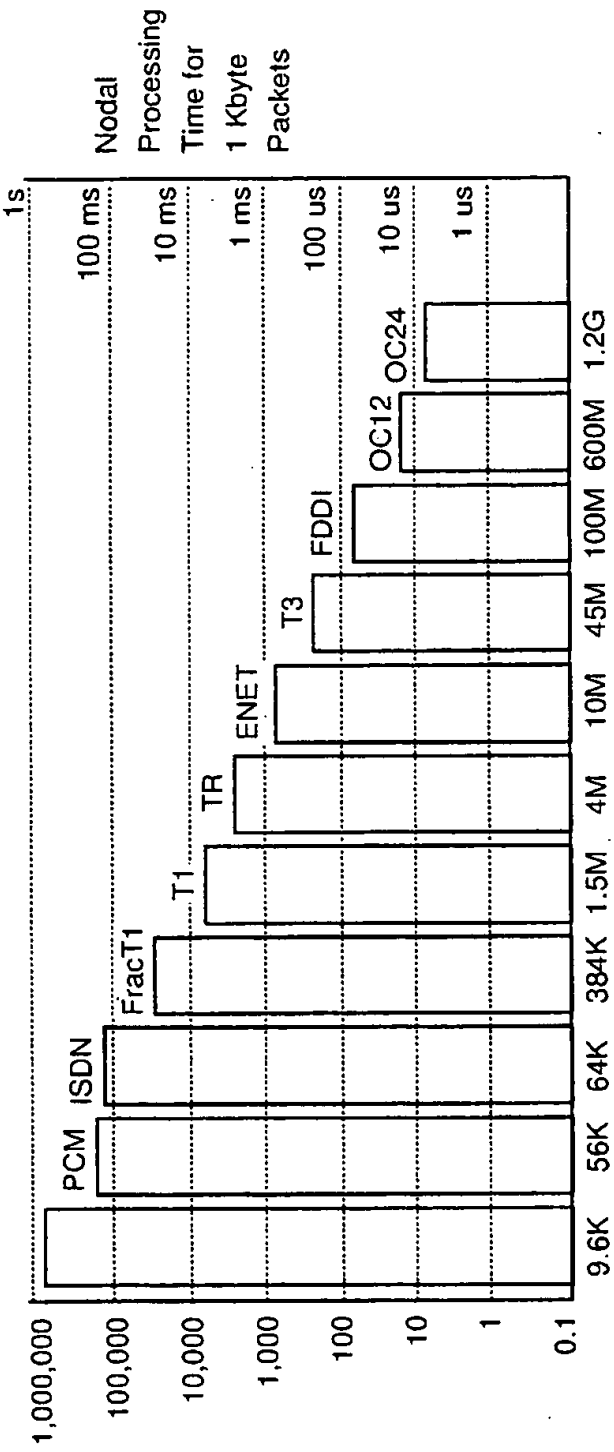

Nodal Processing Time for 1 Kbyte Packets
1s
100 ms
10 ms
1 ms
100 us
10 us
1 us
PCM
ISDN
FracT1
T1
TR
ENET
T3
FDDI
OC12
OC24
9.6K
56K
64K
384K
1.5M
4M
10M
45M
100M
600M
1.2G
Link Speed
1,000,000
100,000
10,000
1,000
100
10
1
0.1
Number of Instructions at 1 Mips

FIG 1

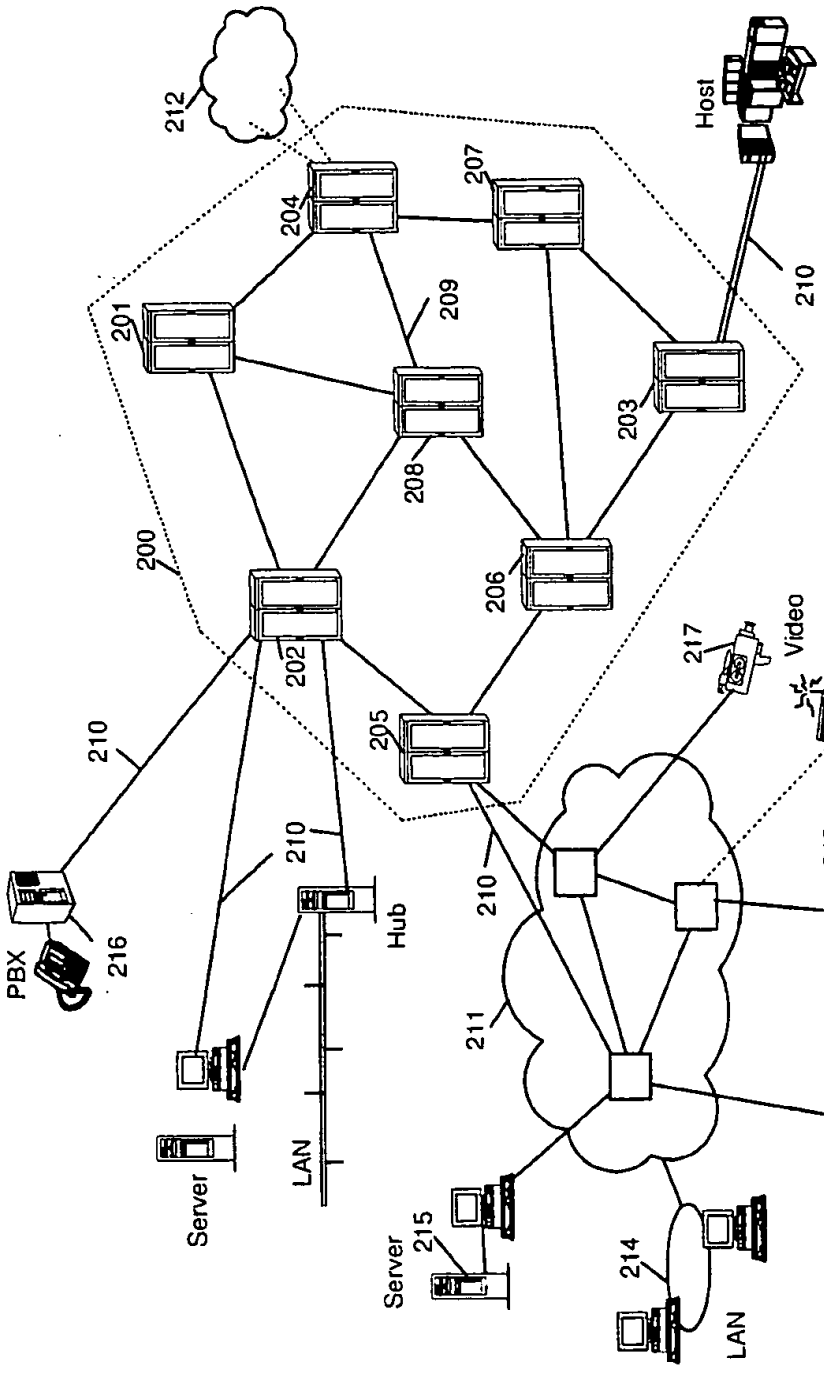

212
Host
204
207
210
201
209
203
208
200
206
202
205
217
Video
Wireless
210
213
210
Hub
210
PBX
216
Host
211
Server
LAN
Server
215
214
LAN
PBX
216

FIG 2

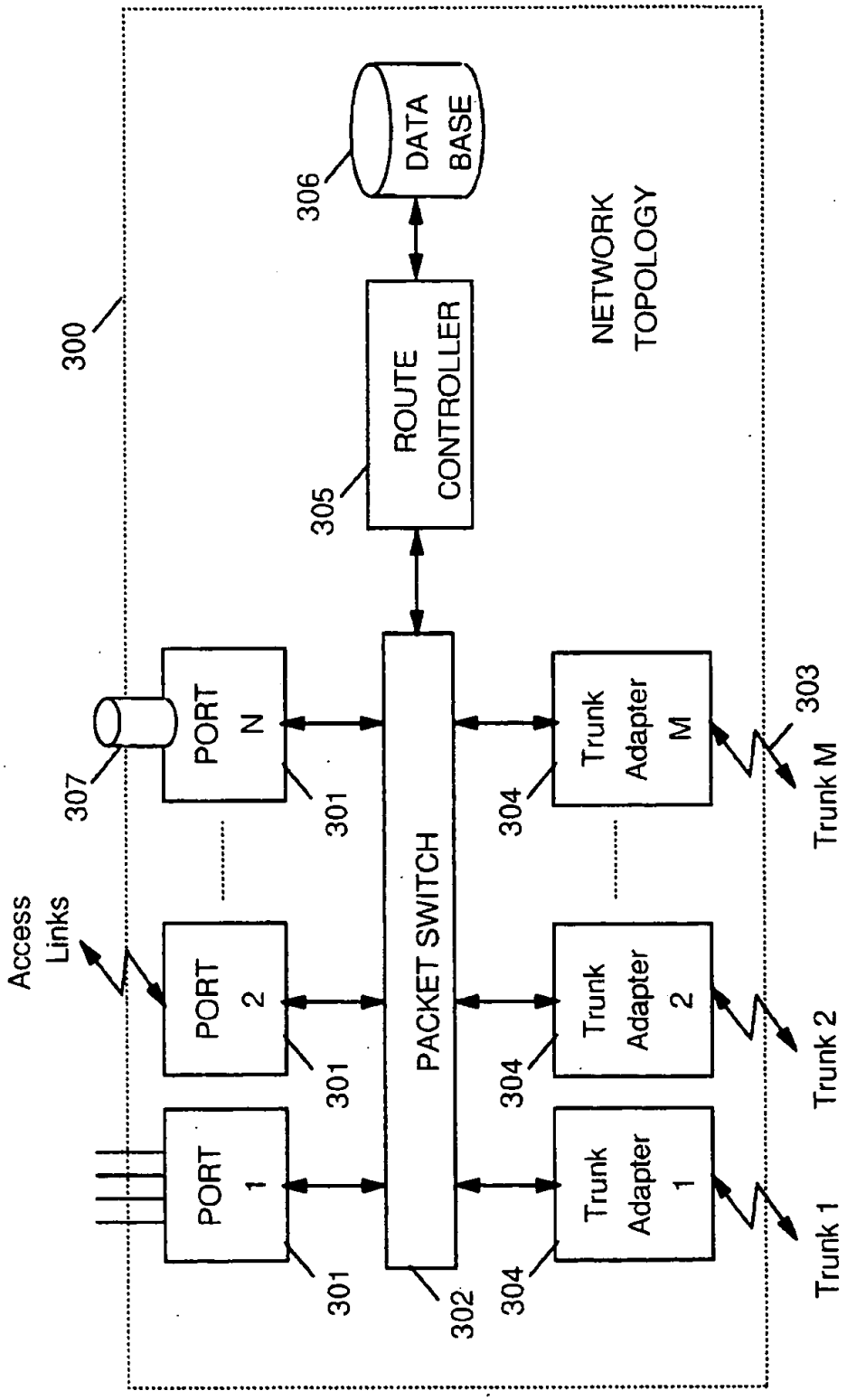

NETWORK TOPOLOGY
300
DATA BASE
306
ROUTE CONTROLLER
305
PACKET SWITCH
302
PORT N
PORT 2
PORT 1
301
307
Access Links
Trunk Adapter M
Trunk Adapter 2
Trunk Adapter 1
304
303
Trunk M
Trunk 2
Trunk 1

FIG 3

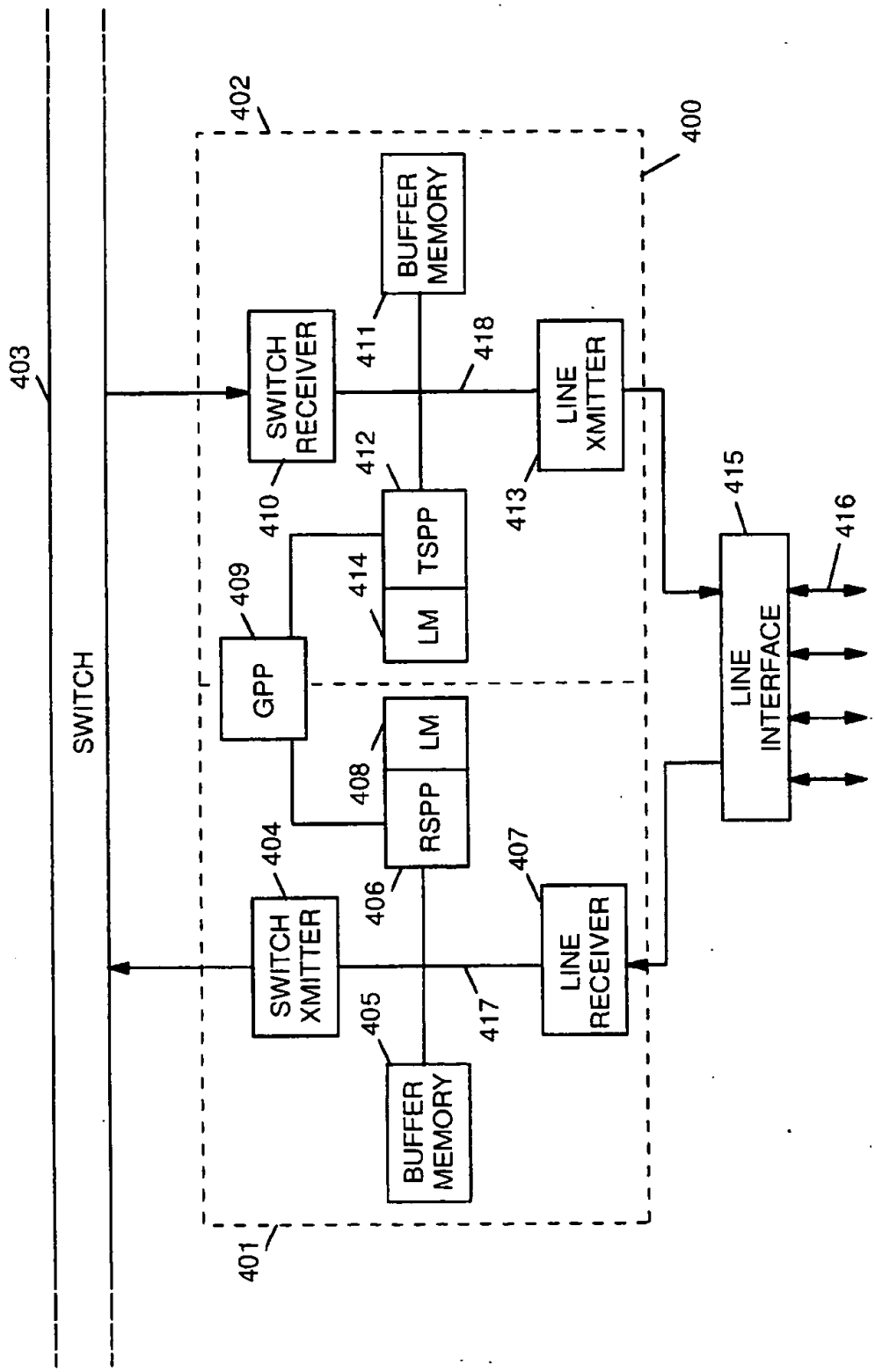

SWITCH
403
402
400
401
BUFFER MEMORY
411
SWITCH RECEIVER
410
418
LINE XMITTER
413
412
TSPP
LM
414
409
GPP
408
LM
RSPP
406
404
SWITCH XMITTER
405
BUFFER MEMORY
407
LINE RECEIVER
417
415
LINE INTERFACE
416

FIG 4

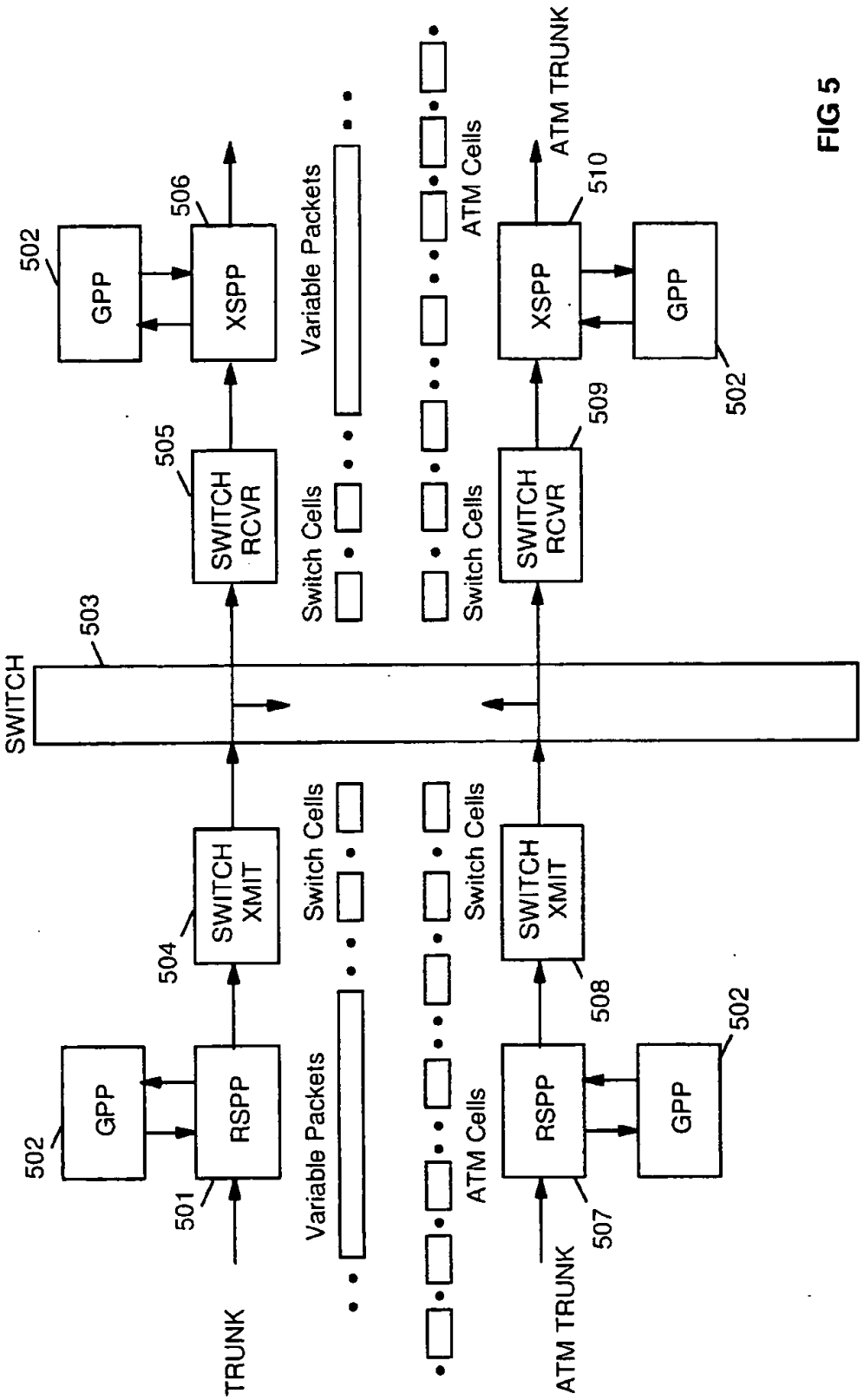

TRUNK
ATM TRUNK
501
507
RSPP
RSPP
502
GPP
GPP
504
508
SWITCH XMIT
SWITCH XMIT
Variable Packets
ATM Cells
Switch Cells
Switch Cells
Switch Cells
ATM Cells
Variable Packets
503
SWITCH
505
509
SWITCH RCVR
SWITCH RCVR
506
510
XSPP
XSPP
ATM TRUNK
502
GPP
GPP
502

FIG 5

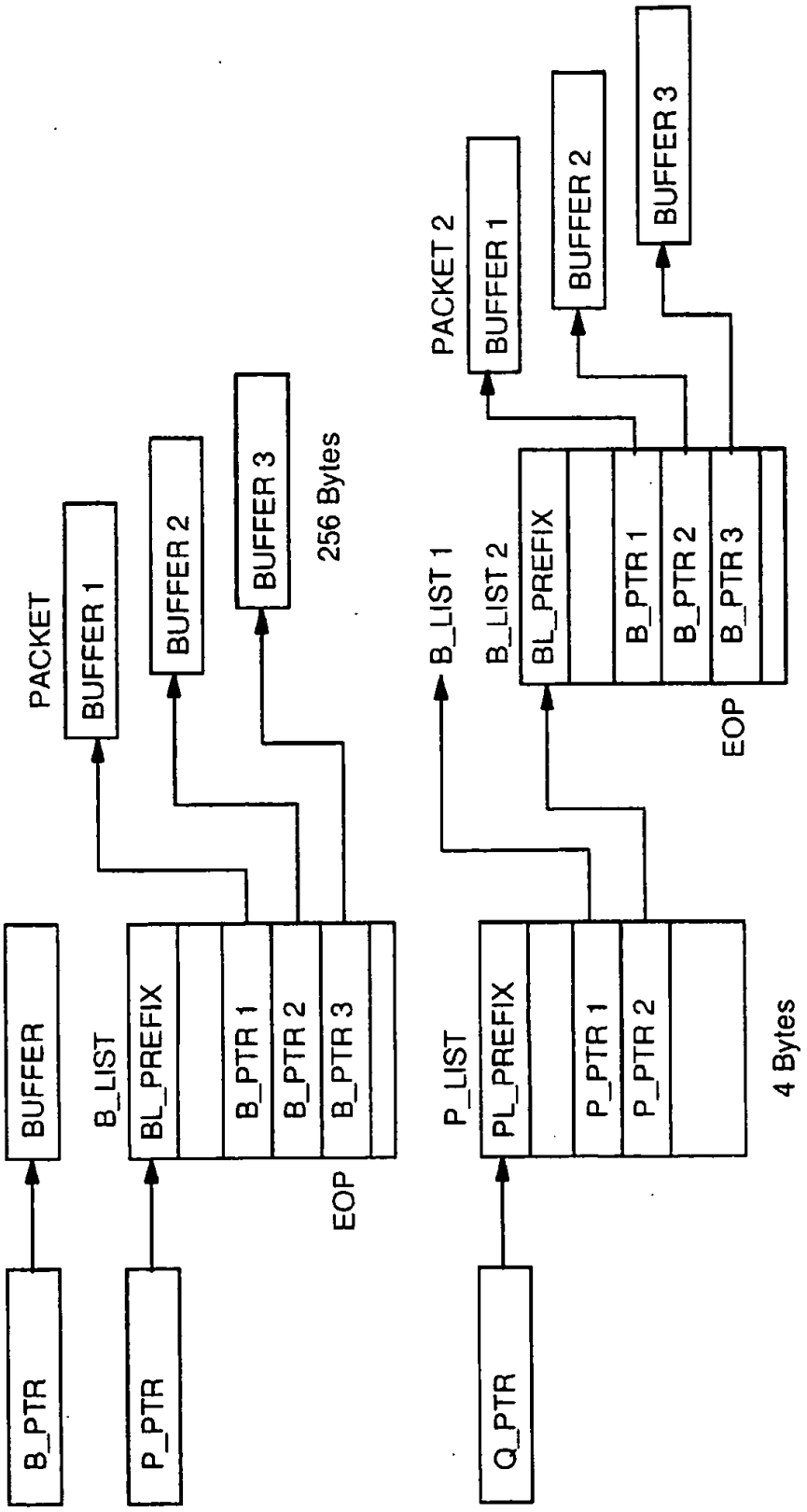

B_PTR
P_PTR
Q_PTR
BUFFER
B_LIST
BL_PREFIX
B_PTR 1
B_PTR 2
B_PTR 3
EOP
PACKET
BUFFER 1
BUFFER 2
BUFFER 3
256 Bytes
P_LIST
PL_PREFIX
P_PTR 1
P_PTR 2
4 Bytes
B_LIST 1
B_LIST 2
PACKET 2
BUFFER 1
BUFFER 2
BUFFER 3
BL_PREFIX
B_PTR 1
B_PTR 2
B_PTR 3
EOP

FIG 6

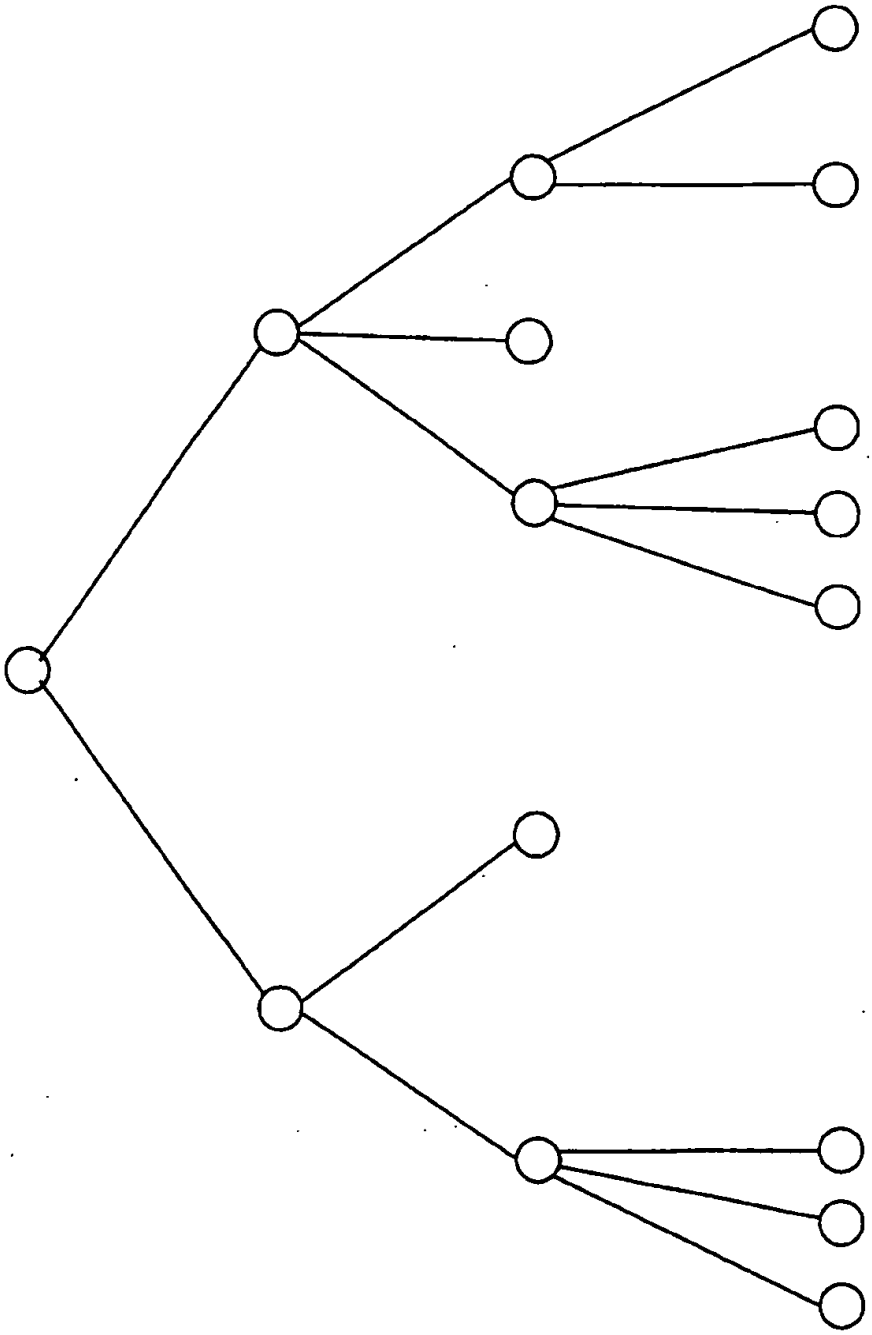


FIG 7

FIG 8A

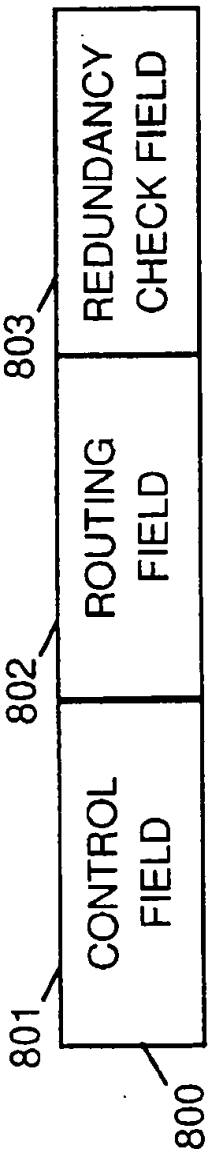
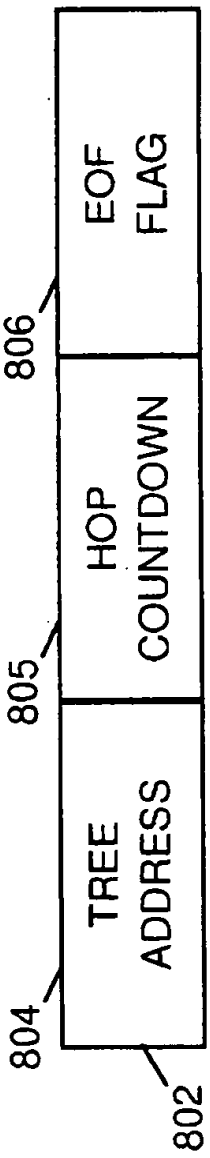

CONTROL FIELD
ROUTING FIELD
REDUNDANCY CHECK FIELD
801
802
803
800

FIG 8B


TREE ADDRESS
HOP COUNTDOWN
EOF FLAG
804
805
806
802

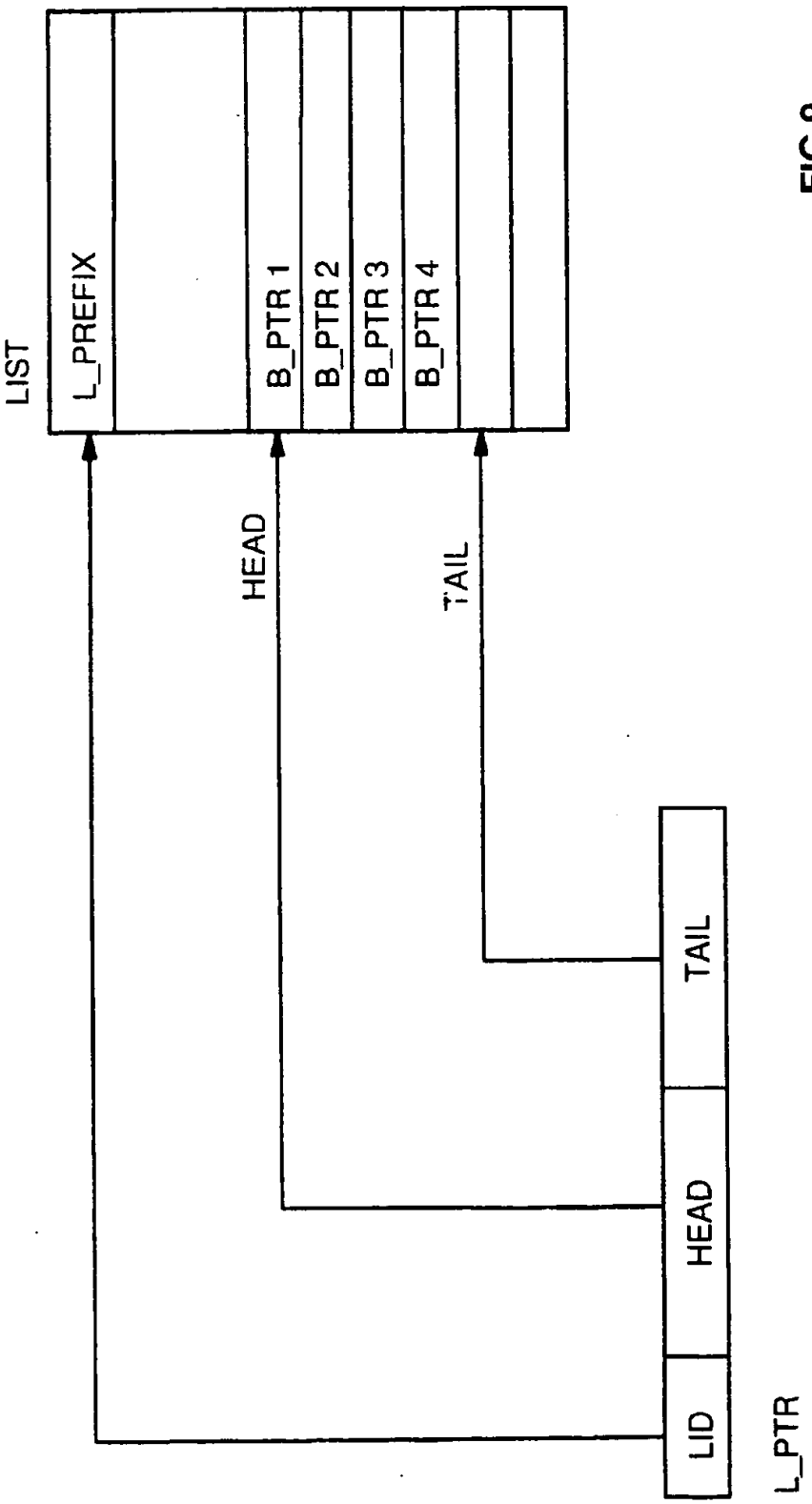

LIST
L_PREFIX
B_PTR 1
B_PTR 2
B_PTR 3
B_PTR 4
HEAD
TAIL
TAIL
HEAD
LID
L_PTR

FIG 9

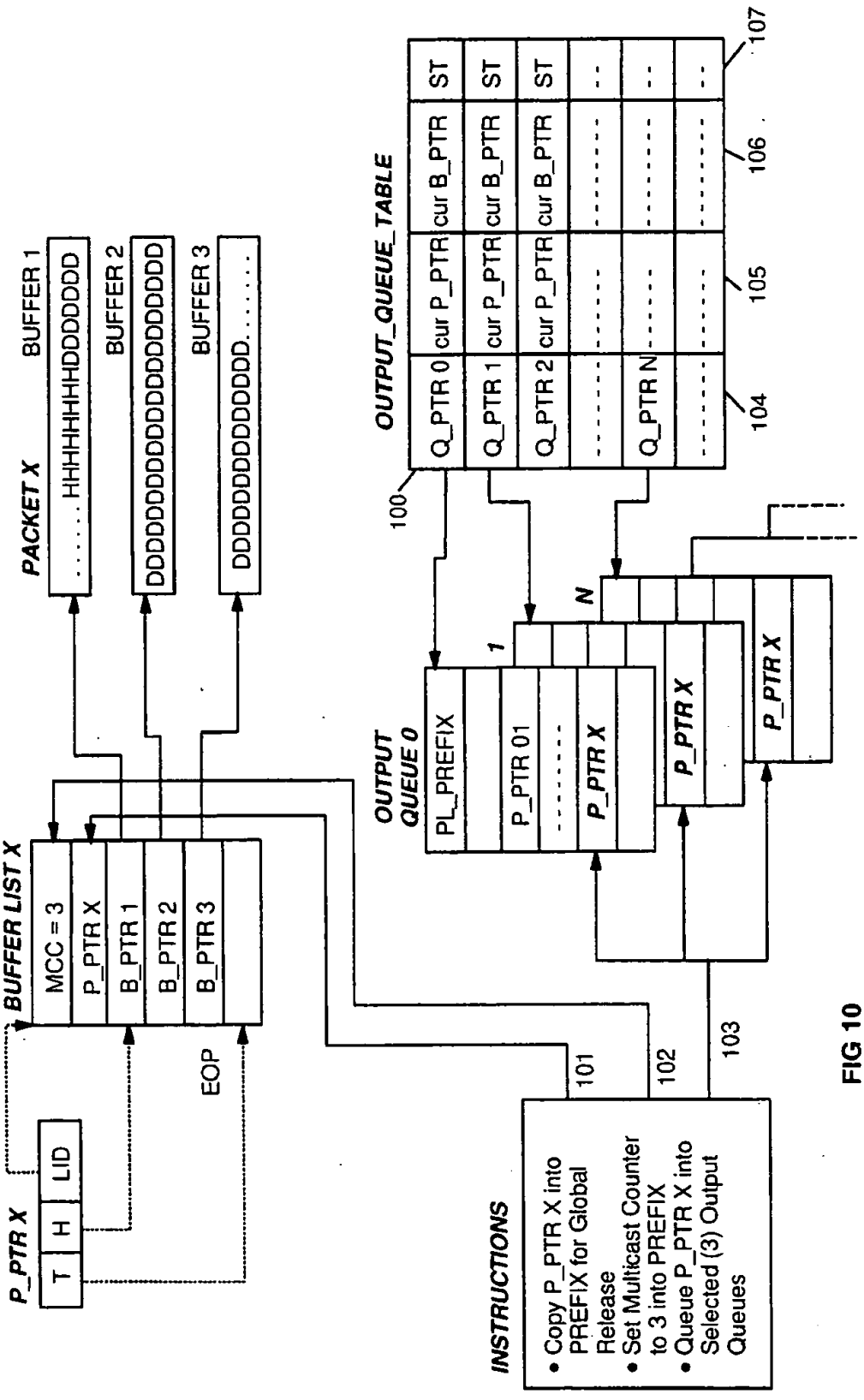

PACKET X
BUFFER 1
HHHHHHHHDDDDDDDD
BUFFER 2
DDDDDDDDDDDDDDDD
BUFFER 3
DDDDDDDDDDDD
BUFFER LIST X
MCC = 3
P_PTR X
B_PTR 1
B_PTR 2
B_PTR 3
EOP
P_PTR X
T
H
LID
OUTPUT_QUEUE_TABLE
Q_PTR 0
Q_PTR 1
Q_PTR 2
Q_PTR N
cur P_PTR
cur B_PTR
ST
100
104
105
106
107
OUTPUT QUEUE 0
PL_PREFIX
P_PTR 01
P_PTR X
1
N
INSTRUCTIONS
• Copy P_PTR X into PREFIX for Global Release
• Set Multicast Counter to 3 into PREFIX
• Queue P_PTR X into Selected (3) Output Queues
101
102
103

FIG 10

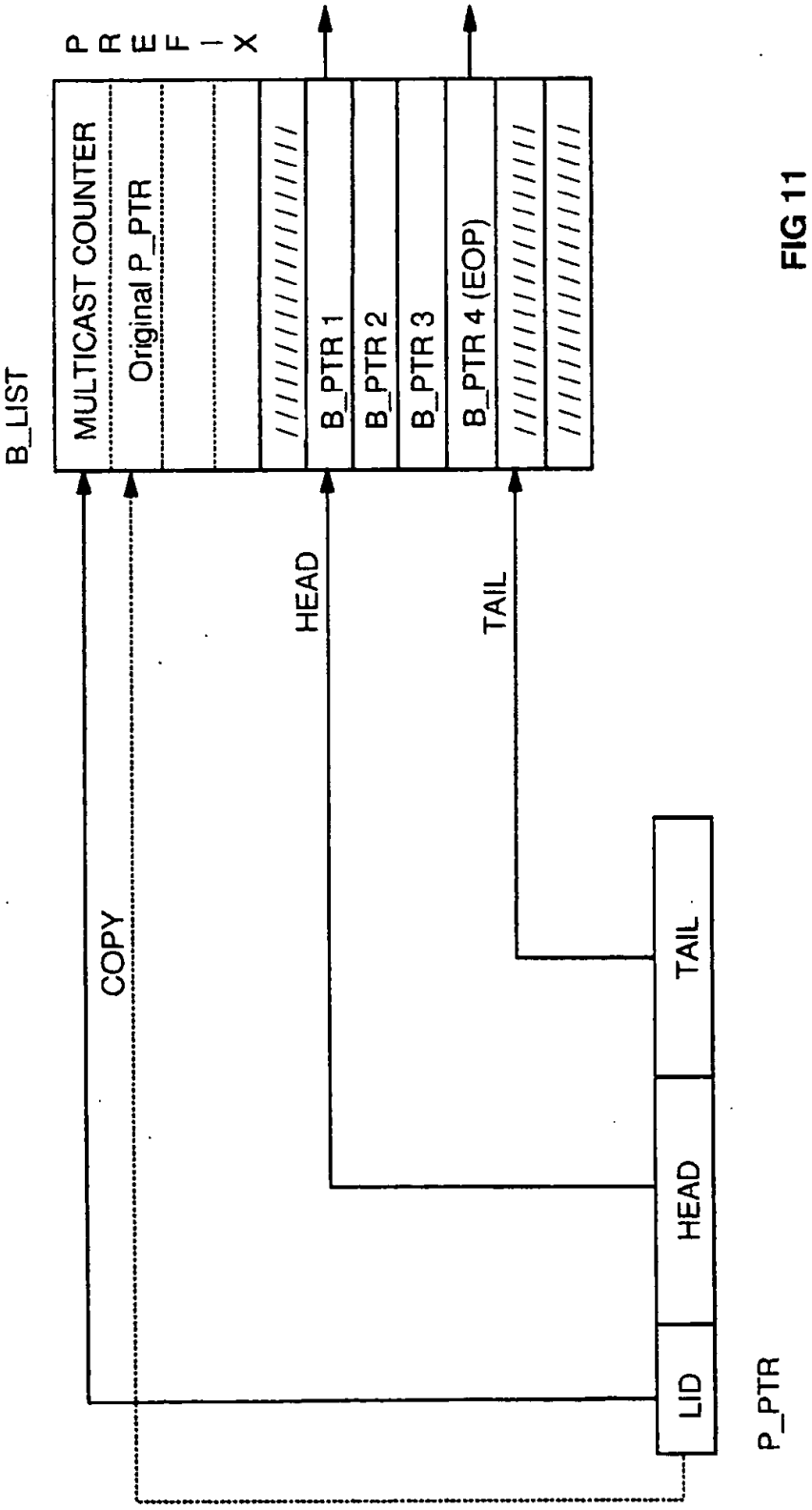

PREF-X
B_LIST
MULTICAST COUNTER
Original P_PTR
B_PTR 1
B_PTR 2
B_PTR 3
B_PTR 4 (EOP)
HEAD
TAIL
COPY
TAIL
HEAD
LID
P_PTR

FIG 11

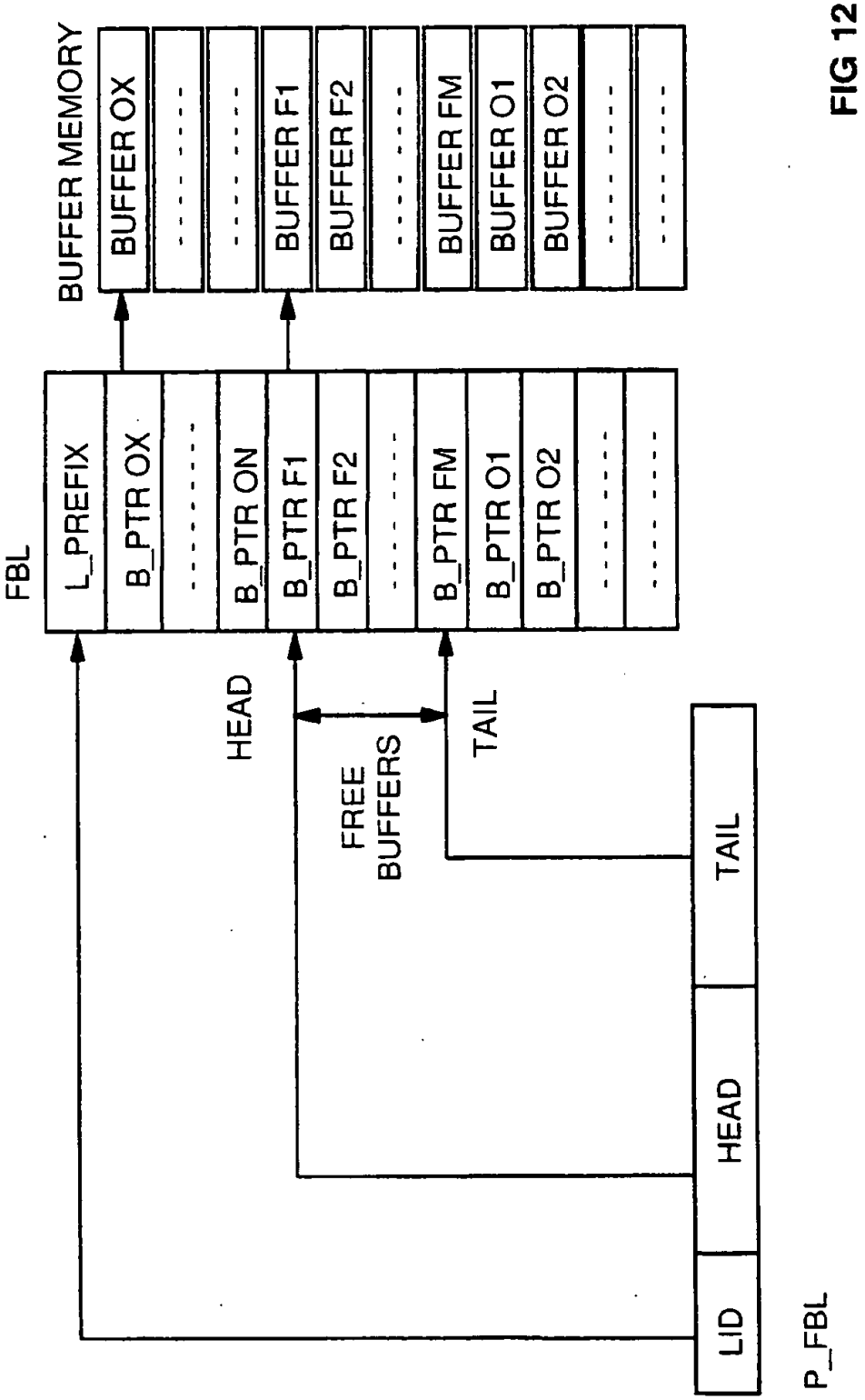

BUFFER MEMORY
BUFFER OX
BUFFER F1
BUFFER F2
BUFFER FM
BUFFER O1
BUFFER O2
FBL
L_PREFIX
B_PTR OX
B_PTR ON
B_PTR F1
B_PTR F2
B_PTR FM
B_PTR O1
B_PTR O2
HEAD
TAIL
FREE BUFFERS
TAIL
HEAD
LID
P_FBL

FIG 12

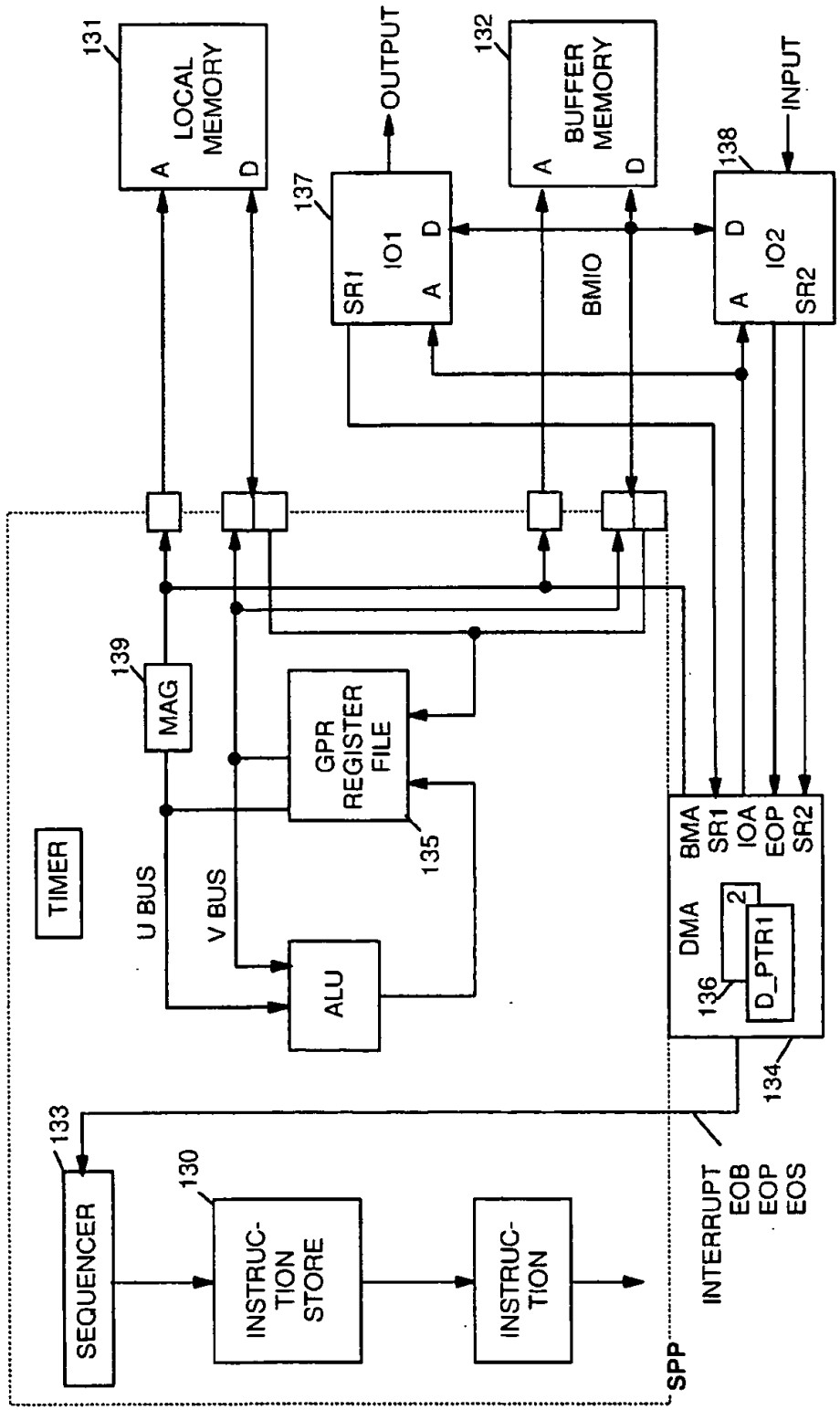

131
LOCAL MEMORY
A
D
OUTPUT
132
BUFFER MEMORY
A
D
INPUT
138
137
SR1
IO1
A
D
BMIO
D
IO2
A
SR2
139
MAG
GPR REGISTER FILE
135
TIMER
U BUS
V BUS
ALU
BMA
SR1
IOA
EOP
SR2
DMA
2
D_PTR1
136
134
133
SEQUENCER
130
INSTRUC-TION STORE
INSTRUC-TION
INTERRUPT
EOB
EOP
EOS
SPP

FIG 13

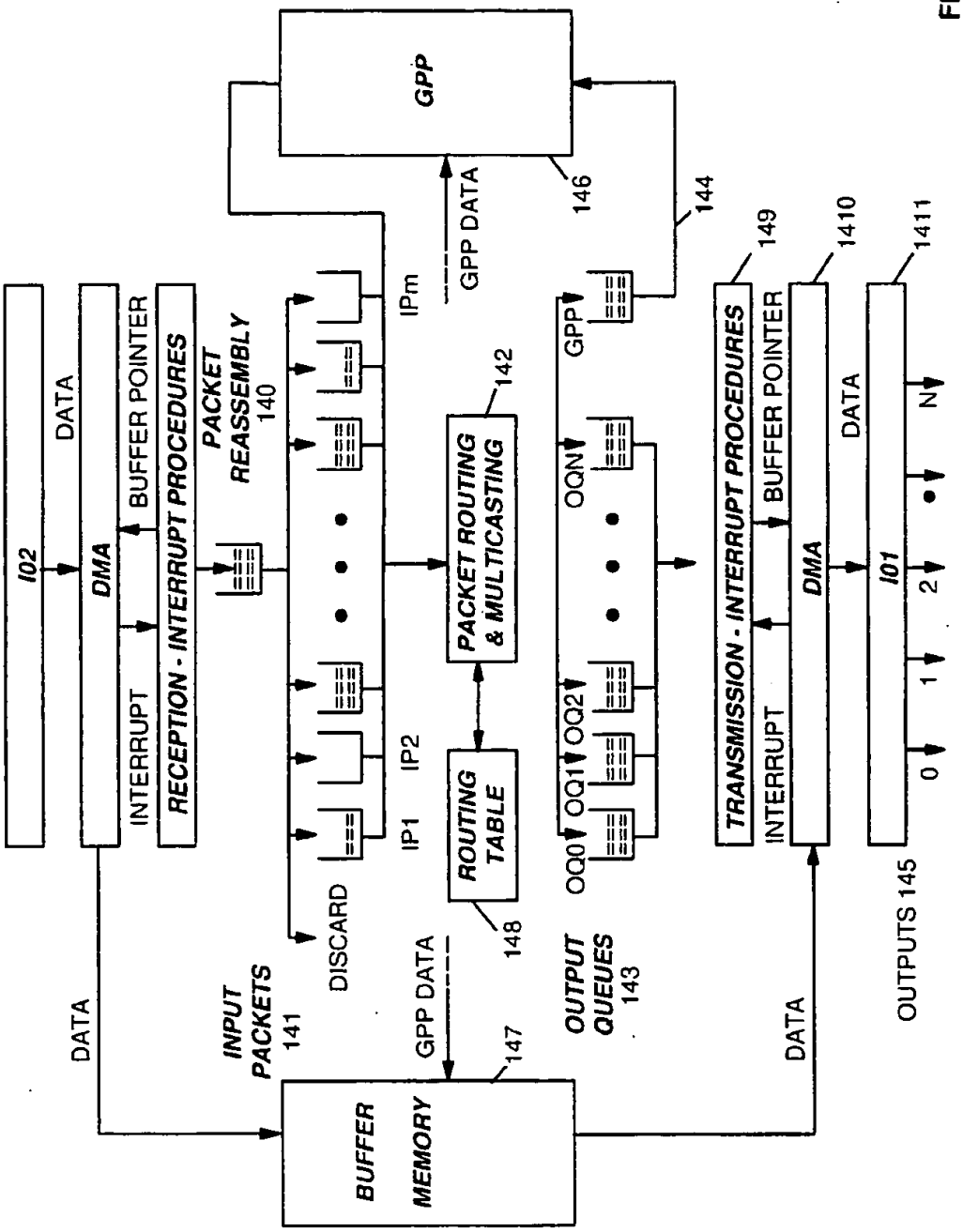

GPP
146
144
GPP DATA
IPm
I02
DATA
DMA
BUFFER POINTER
INTERRUPT
RECEPTION - INTERRUPT PROCEDURES
PACKET REASSEMBLY
140
142
PACKET ROUTING & MULTICASTING
ROUTING TABLE
148
IP1
IP2
DISCARD
INPUT PACKETS
141
GPP DATA
147
BUFFER MEMORY
DATA
OUTPUT QUEUES
143
OQ0
OQ1
OQ2
OQN
GPP
TRANSMISSION - INTERRUPT PROCEDURES
149
BUFFER POINTER
INTERRUPT
DMA
1410
DATA
I01
1411
0
1
2
N
OUTPUTS 145

FIG 14

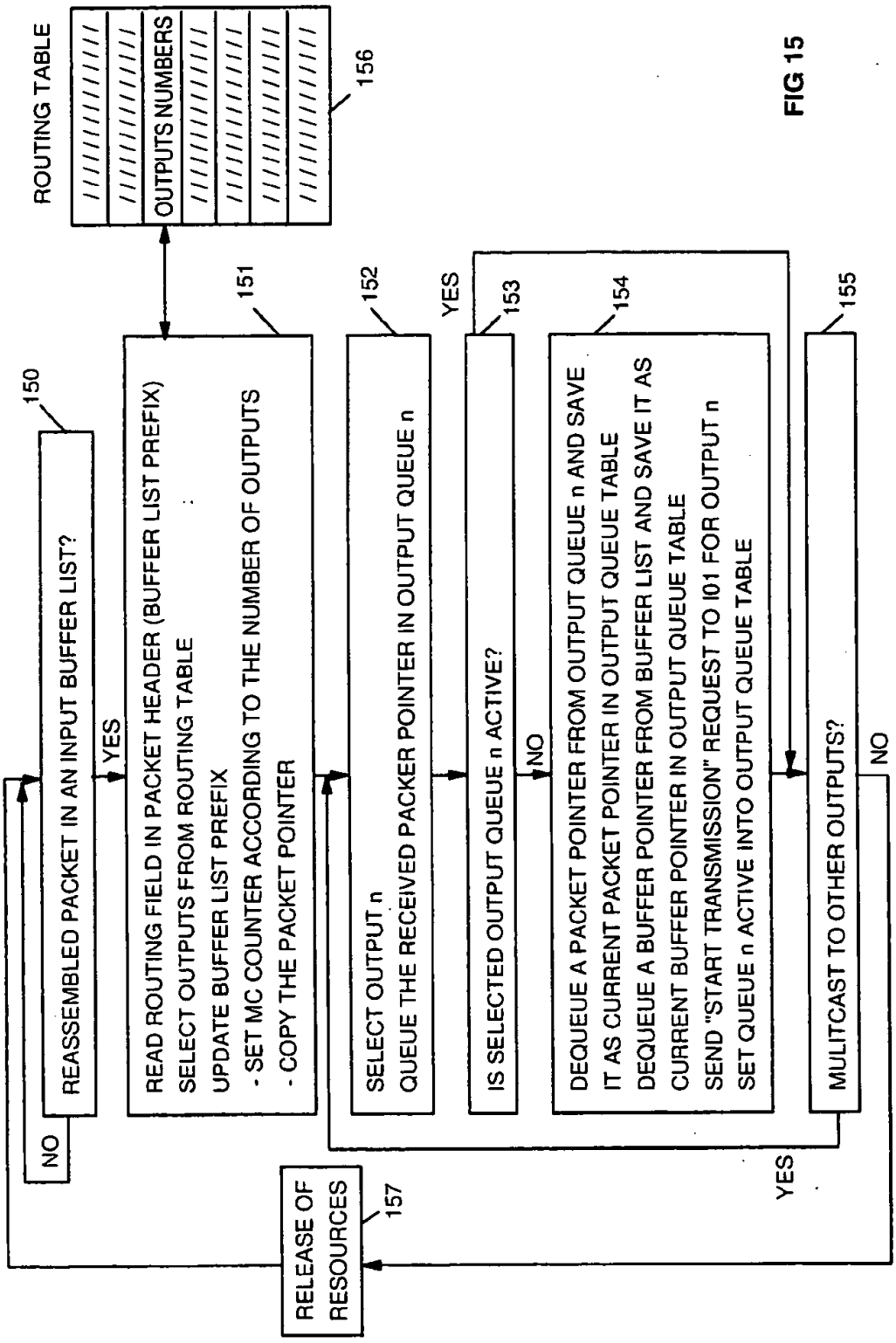

ROUTING TABLE
OUTPUTS NUMBERS
156
150
REASSEMBLED PACKET IN AN INPUT BUFFER LIST?
NO
YES
151
READ ROUTING FIELD IN PACKET HEADER (BUFFER LIST PREFIX)
SELECT OUTPUTS FROM ROUTING TABLE
UPDATE BUFFER LIST PREFIX
- SET MC COUNTER ACCORDING TO THE NUMBER OF OUTPUTS
- COPY THE PACKET POINTER
152
SELECT OUTPUT n
QUEUE THE RECEIVED PACKER POINTER IN OUTPUT QUEUE n
YES
153
IS SELECTED OUTPUT QUEUE n ACTIVE?
NO
154
DEQUEUE A PACKET POINTER FROM OUTPUT QUEUE n AND SAVE IT AS CURRENT PACKET POINTER IN OUTPUT QUEUE TABLE
DEQUEUE A BUFFER POINTER FROM BUFFER LIST AND SAVE IT AS CURRENT BUFFER POINTER IN OUTPUT QUEUE TABLE
SEND "START TRANSMISSION" REQUEST TO I01 FOR OUTPUT n
SET QUEUE n ACTIVE INTO OUTPUT QUEUE TABLE
155
MULITCAST TO OTHER OUTPUTS?
YES
NO
RELEASE OF RESOURCES
157

FIG 15

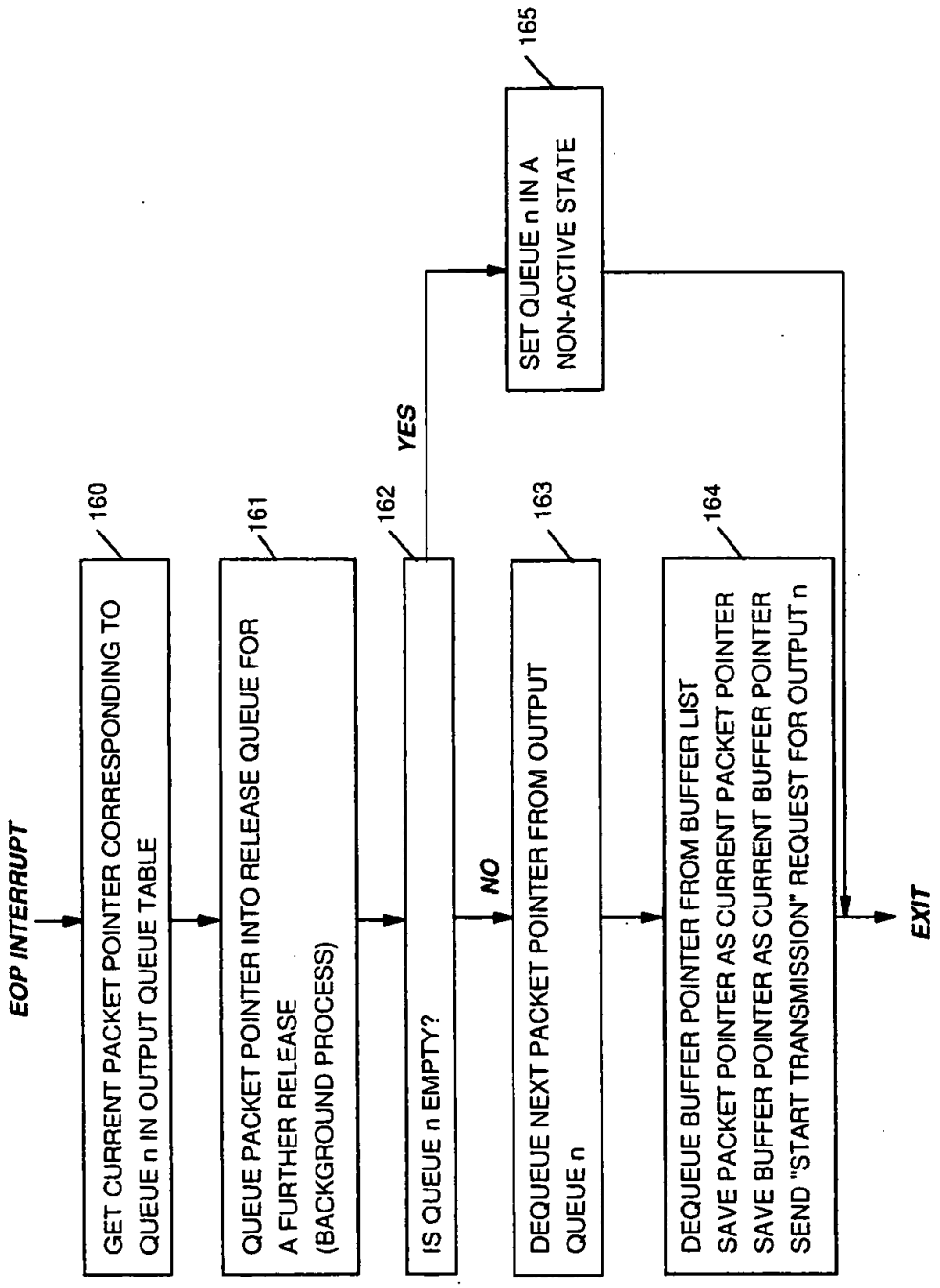
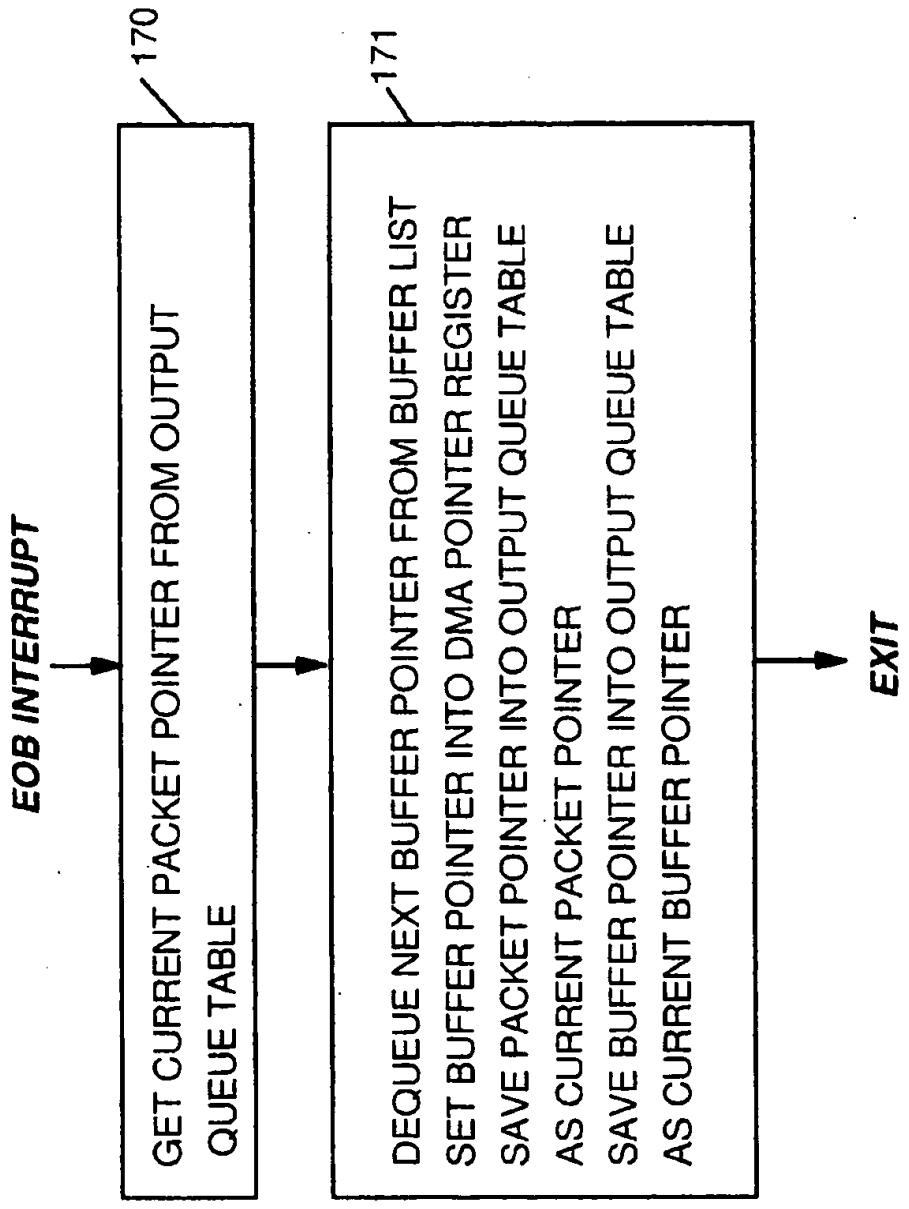

EOP INTERRUPT
160
GET CURRENT PACKET POINTER CORRESPONDING TO QUEUE n IN OUTPUT QUEUE TABLE
161
QUEUE PACKET POINTER INTO RELEASE QUEUE FOR A FURTHER RELEASE (BACKGROUND PROCESS)
162
IS QUEUE n EMPTY?
YES
NO
165
SET QUEUE n IN A NON-ACTIVE STATE
163
DEQUEUE NEXT PACKET POINTER FROM OUTPUT QUEUE n
164
DEQUEUE BUFFER POINTER FROM BUFFER LIST
SAVE PACKET POINTER AS CURRENT PACKET POINTER
SAVE BUFFER POINTER AS CURRENT BUFFER POINTER
SEND "START TRANSMISSION" REQUEST FOR OUTPUT n
EXIT

FIG 16

FIG 17


EOB INTERRUPT
170
GET CURRENT PACKET POINTER FROM OUTPUT QUEUE TABLE
171
DEQUEUE NEXT BUFFER POINTER FROM BUFFER LIST
SET BUFFER POINTER INTO DMA POINTER REGISTER
SAVE PACKET POINTER INTO OUTPUT QUEUE TABLE AS CURRENT PACKET POINTER
SAVE BUFFER POINTER INTO OUTPUT QUEUE TABLE AS CURRENT BUFFER POINTER
EXIT

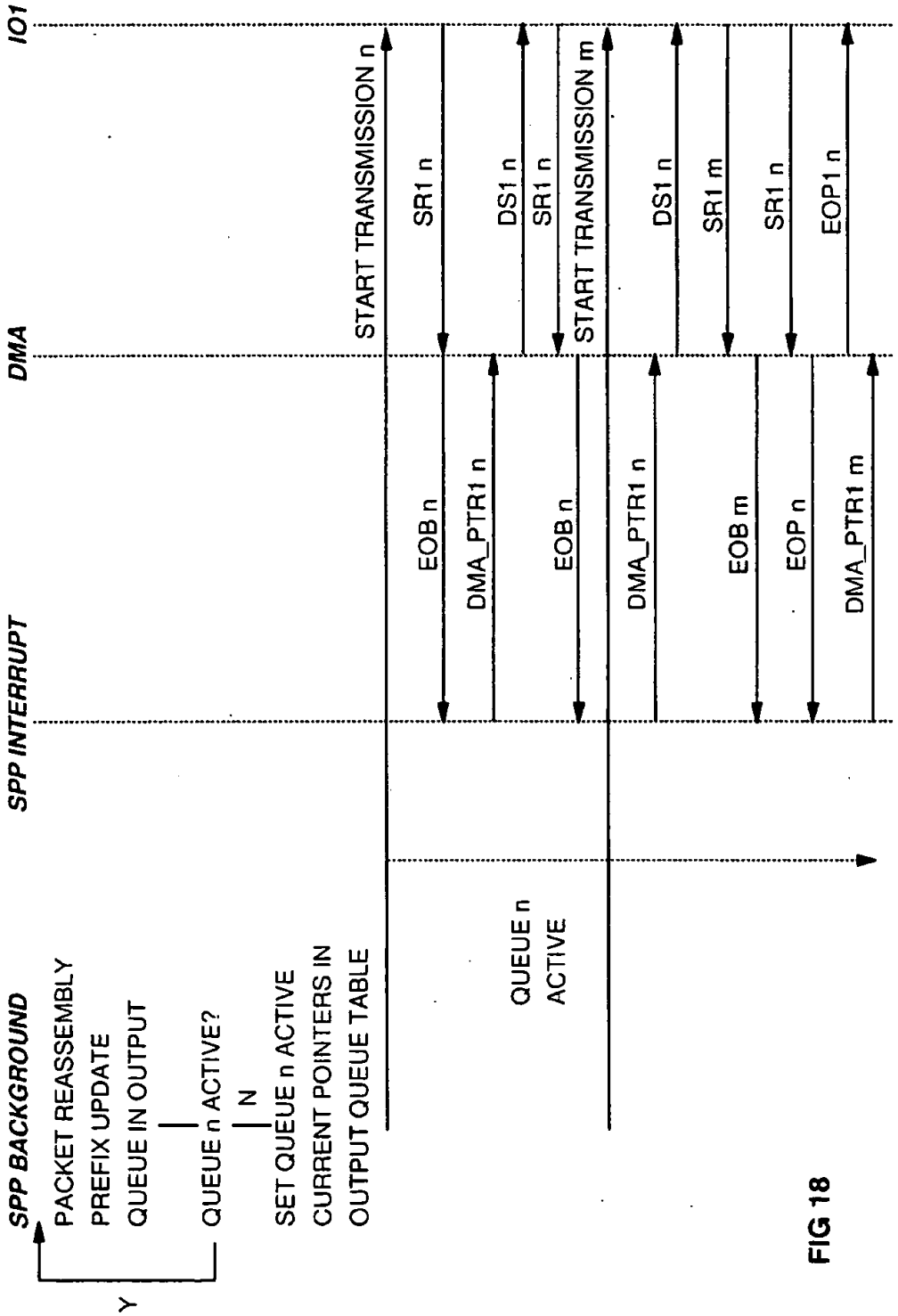

IO1
DMA
SPP INTERRUPT
SPP BACKGROUND
START TRANSMISSION n
SR1 n
DS1 n
SR1 n
START TRANSMISSION m
DS1 n
SR1 m
SR1 n
EOP1 n
EOB n
DMA_PTR1 n
EOB n
DMA_PTR1 n
EOB m
EOP n
DMA_PTR1 m
QUEUE n ACTIVE
PACKET REASSEMBLY
PREFIX UPDATE
QUEUE IN OUTPUT
QUEUE n ACTIVE?
Y
N
SET QUEUE n ACTIVE
CURRENT POINTERS IN
OUTPUT QUEUE TABLE

FIG 18

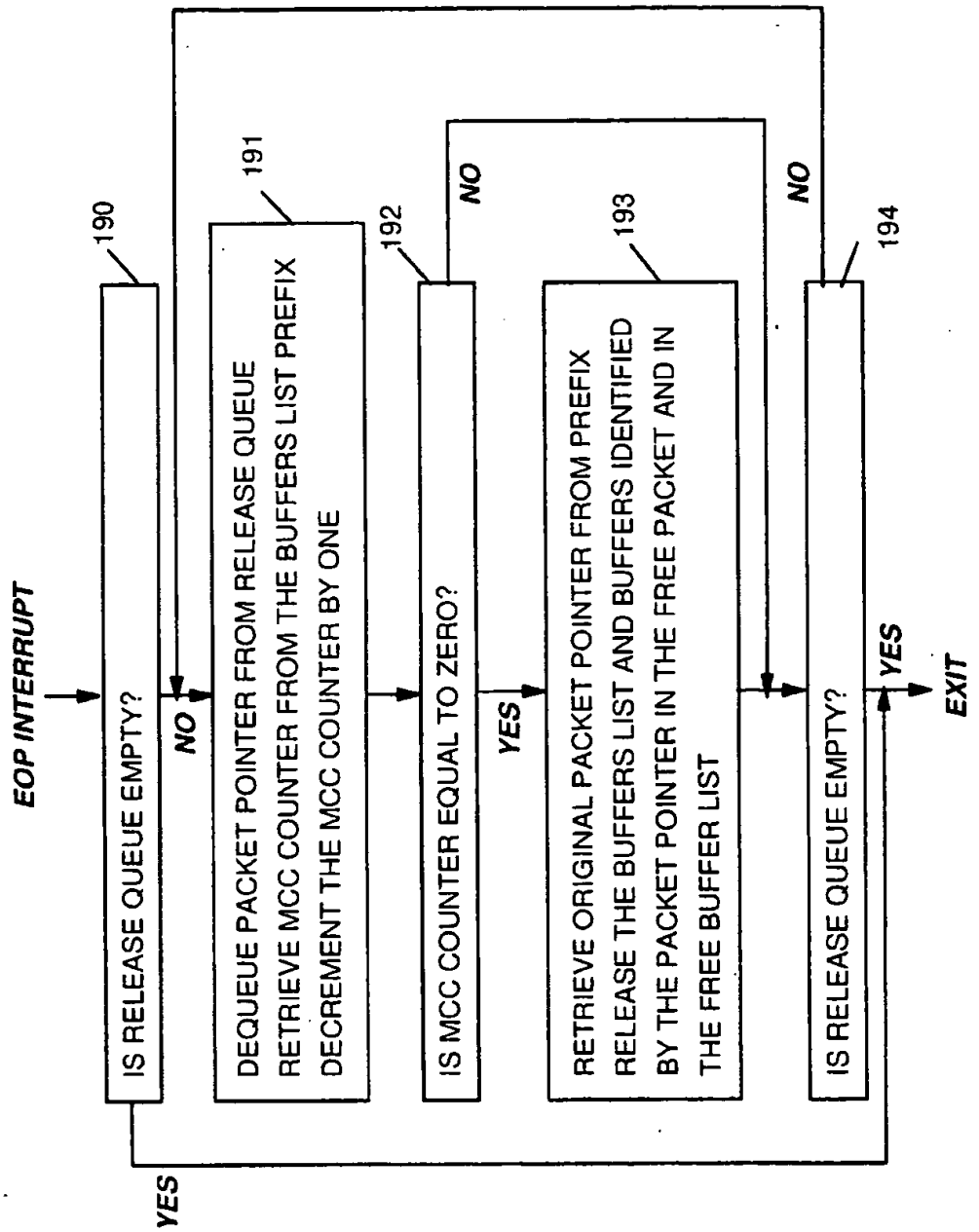

EOP INTERRUPT
190
IS RELEASE QUEUE EMPTY?
NO
191
DEQUEUE PACKET POINTER FROM RELEASE QUEUE
RETRIEVE MCC COUNTER FROM THE BUFFERS LIST PREFIX
DECREMENT THE MCC COUNTER BY ONE
192
IS MCC COUNTER EQUAL TO ZERO?
NO
YES
193
RETRIEVE ORIGINAL PACKET POINTER FROM PREFIX
RELEASE THE BUFFERS LIST AND BUFFERS IDENTIFIED
BY THE PACKET POINTER IN THE FREE PACKET AND IN
THE FREE BUFFER LIST
194
IS RELEASE QUEUE EMPTY?
NO
YES
EXIT
YES

FIG 19

EFFICIENT POINT-TO-POINT AND MULTI-POINT ROUTING MECHANISM FOR PROGRAMMABLE PACKET SWITCHING NODES IN HIGH SPEED DATA TRANSMISSION NETWORKS

TECHNICAL FIELD

The present invention relates to an efficient point-to-point and multi-points routing system and method for programmable data communication adapters in packet switching nodes of high speed networks.

BACKGROUND ART

The telecommunication environment is in full evolution and has changed considerably this recent years. The principal reason has been the spectacular progress realized in the communication technology:

the maturing of fiber optical transmission. High speed rates can now be sustained with very low bit error rates.

the universal use of digital technologies within private and public telecommunications networks.

In relation with these new emerging technologies, the offer of the telecommunication companies, public or private, are evolving:

The emergence of high speed transmissions entails an explosion in the high bandwidth connectivity.

the increase of the communication capacity generates more attractive tariffs.

A higher flexibility is offered to the users to manage their growth through a wide range of connectivity options, an efficient bandwidth management and the support of new media.

Once sampled and digitally encoded, voice, video and image derived data can be merged with pure data for a common and transparent transport.

Abundant, cheap communications means that many potential applications that where not possible before because of cost are now becoming attractive. In this environment, three generic requirements are expressed by the users:

Doing old applications better,

Optimizing communication networks,

Doing new applications.

High Performance Networks

In a first step, T1 backbone networks were primarily deployed with TDM (Time Division Multiplexing) technology to achieve cost savings through line aggregation. These systems easily supported the fixed bandwidth requirements of host/terminal computing and 64 Kbps PCM (Pulse Code Modulation) voice traffic.

The data transmission is now evolving with a specific focus on applications and by integrating a fundamental shift in the customer traffic profile. Driven by the growth of workstations, the local area networks (LAN) interconnection, the distributed processing between workstations and super computers, the new applications and the integration of various and often conflicting structures—hierarchical versus peer to peer, wide (WAN) versus local (LAN) area networks, voice versus data—the data profile has become higher in bandwidth, bursting, non deterministic and requires more connectivity. Based on the above, it is clear that there is strong requirement to support distributed computing applications across high speed backbones that may be carrying

LAN traffic, voice, video, and traffic among channel attached hosts, business workstations, engineering workstations, terminals, and small to intermediate file servers. This traffic reflects a heterogeneous mix of:

end user network protocols including Ethernet, Tocken Ring, APPN, FDDI, OSI, ISDN, ATM . . . , and

real time (steady stream traffic such as voice and video) and non real time (bursty nature traffic such as interactive data) transmissions.

This vision of a high speed protocol-agile backbone network is the driver for the emergence of fast packet switching networks architectures in which data, voice, and video information is digitally encoded, chopped into small packets and transmitted through a common set of nodes and links. Although low speed links may exist, the availability of fiber optic links will make cost effective to have a few links of high speed rather that many links of low speed. In addition to the high speed backbone, there exists a peripheral network which essentially provides access to the switching nodes. This peripheral network is composed of relatively low speed links which may not use the same protocols or switching techniques used in the backbone. In addition, the peripheral network performs the task of multiplexing the relatively slow end users traffic to the high speed backbone. Thus, backbone switching nodes are principally handling high speed lines. The number of high speed links entering each switching node is relatively small but the aggregate throughput very high in the Giga-bits per second range.

Throughput

The key requirement of these new architectures is to reduce the end-to-end delay in order to satisfy real time delivery constraints and to achieve the necessary high nodal throughput for the transport of voice and video. Increases in link speeds have not been matched by proportionate increases in the processing speeds of communication nodes and the fundamental challenge for high speed networks is to minimize the packet processing time within each node. As example, for meeting a typical 100 ms delay to deliver a voice packet between two end users:

A total of 36 ms might be needed for the packetization and play-out functions at the end points.

About 20 ms is the unalterable propagation delay needed, say, to cross the United States.

There remains 44 ms for all the intra-node processing time as the packet moves through the network. In a 5 nodes network, each node would have about 8 ms for all processing time including any queueing time. In a 10 nodes network, each node would have about 4 ms.

Another way of looking the same constraint is illustrated in FIG. 1: taking a node with an effective processing rate of 1 MIPS (Millions of Instructions Per Second), it is possible to fill a 9.6 kbps line with 1000 byte packets even if a network node must execute 833 000 instructions per packet processed. For a 64 kbps line the node can afford 125 000 instructions per packet. In order to fill an OC24 link, however, our 1 MIPS node could only execute 7 instruction per packet! In the latter case even an effective rate of 10–30 MIPS would allow only 70–200 instructions per packet.

In order to minimize the processing time and to take full advantage of the high speed/low error rate technologies, most of the transport functions provided by the new high bandwidth network architectures are performed on an end-to-end basis. This includes the flow control and error recov-

ery for data, the packetization and reassembly for voice and video. The protocol is simplified:

First, there is no need for transit node to be aware of individual (end user to end user) transport connections.

Secondly high performance and high quality links does not require any more node to node error recovery or retransmission. Congestion and and flow control are managed at the access and end points of the network connections reducing both the awareness and the function of the intermediate nodes.

Packet size

Transmission of real time data, like voice or video packets, which must be delivered to the receiver at a steady, uniform rate (isochronous mode) requires the use of short packets. In another side, pure data does not have any problem with transit delay. They are generated in a very bursty and non deterministic manner. The longer packet are, the fewer packets per second must be switched for a given data throughput. In order to take full advantage of the different data packet transmission systems, the data transfer across the network must be done with packets of nearly the same size as the user packets without processing them into artificial lengths. As opposed to solely data networks or solely voice or video networks, the high speed network architectures have to support a plurality of heterogeneous transmission protocols operating with variable length packets.

Connectivity

In a high speed network, the nodes must provide a total connectivity. This includes attachment of the customer's devices, regardless of vendor or protocol, and the ability to have the end user communicate with any other device. Traffic types include data, voice, video, fax, graphic, image. The node must be able to take advantage of all common carrier facilities and to be adaptable to a plurality of protocols: all needed conversions must be automatic and transparent to the end user. For example, a high speed node must not have any dependencies on the existence of SNA (System Network Architecture) equipments on a user network. It has to be able to offer a similar level of service in a SNA environment as in a non-SNA environment made of routers, Private Branch eXchanges (PBXs), Local Area Networks (LAN) . . . .

Key Requirements

The efficient transport of mixed traffic streams on very high speed lines means for each communication node of the network a set of requirements in term of performance and resource consumption which can be summarized as follows:

a very short packet processing time,

a very high throughput,

an efficient queue and buffer management,

a limited number of instructions per packet,

a minimum impact of the control flow on the user traffic, and

a very large flexibility to support a wide range of connectivity options.

The high bandwidth dictates the need of specialized hardware to support very fast packet handling and control protocols, and to satisfy the real time transmission needs of the voice and video traffic. The processing time being the main bottleneck in high speed networks, most of the communication nodes today are built around high speed switching hardware to off-load the routing packet handling and routing functions from the processor.

However, on equal performances, a software approach represents the most adequate solution for each node to meet the connectivity and flexibility requirements and to optimize the manufacturing and adaptation costs. The line adapters, are based on a common hardware design and are configured by means of a specific programming to execute either the access point or inter nodal transport functions. The adaptability of the adapters to support different access protocols and data streams—Frame Relay, HDLC (High level Data Link-Control), CBO (Continuous Bit Operations), ATM (Asynchronous Transfer Mode), . . .—is provided by logical components called Access Agents. Such logical associations Adapter/Access Agent are specified by software, providing a very large flexibility at a reduced cost. Each line adapter is automatically configured at system initiation according to:

the adapter function, and

the access protocol.

Programmable High Performance Communication Nodes

The throughput of a communication adapter is defined as the total time required to handle a data packet from the input to the output. However, the packet size being application dependent, two measures are currently used to evaluate the performances of adapters:

first, the number of packets of fixed length the adapter is able to handle in a second (packet throughput),

second, the number of bits per second the adapter is able to transmit in case of infinite packet length (data throughput).

The performances depend on the hardware and on the processor capabilities but the main throughput limiting factor is the packet processing time and this processing time is directly related to the number of instructions required to handle a packet. The operations on the packets can be divided in two categories:

the background process with the operations of packet routing, assembling—disassembling, formatting, bandwidth management, priority, ... This process is designed according to the adapter function, but in the same application, the background process is identical for all packets independently of their size.

the buffering process with the interrupt routines. This operation is generic for all adapter types. The processing time required for this operation is directly proportional to the packet length.

The interrupt routines are the way real time is supported by the processor. They must be as short as possible to avoid the overrun on the input device and the underrun on the output device. They commands the performance of the adapter in term data throughput (bits per second). For packets of infinite length, the background process disappears and the throughput bottleneck is the interrupt response time which depends of the queuing and dequeueing operations.

In another way, to maximize the packet throughput (number of packets per seconds that the adapter is able to transmit), the number of instructions required by the background process must be reduced to a minimum.

Non published European patent application 93480087.1 (IBM docket FR993027) entitled *Programmable High Per-*

formance Data Communication Adapter for Packet Transmission Networks (prior art under Article 54(3) EPC), which content is herein incorporated by simple reference, discloses a high performance packet buffering method and system in a programmable data communication adapter. The adapter includes a programmable processor, for receiving and transmitting data packets of fixed or variable length. The system is designed to optimize the queueing and dequeueing operations and in particular to minimize the number of instructions to manipulate the packets. It is characterized in that it comprises:

means for storing data packets in buffers,

means for identifying said data packets in the buffers

means for queueing in a local memory the packet identifiers in a single instruction,

means for dequeueing from the local memory the packet identifiers in another single instruction,

means for releasing the buffers.

Each instruction comprises up to three operations executed in parallel by said processor:

an arithmetical and logical (ALU) operation on the packet identifiers,

a memory operation on the local memory, and

a sequence operation.

SUMMARY OF THE INVENTION

An efficient point-to-point and multi-points routing system and method for data communication adapters in packet switching nodes of a high speed network is disclosed. The general principles of the present invention can be summarized as follows:

data packets are never copied during their routing through the adapter, only packet pointers are copied for each destination,

each output is processed independently on a priority mode,

no overhead generated by the multi-points routing in the interrupt procedures,

release of the resources on a non priority mode (background procedure).

DESCRIPTION OF THE DRAWINGS

FIG. **1** shows the processing times (or number of instructions per second) required in function of the different line throughputs supported by the present invention.

FIG. **2** shows a typical model of high speed packet switching network including the access and transit nodes claimed in the present invention.

FIG. **3** describes a high speed Routing Point according to the present invention.

FIG. **4** shows a programmable high performance adapter according to the present invention.

FIG. **5** represents the receive and transmit data flows in a Trunk Adapter.

FIG. **6** illustrates the buffer, packet and queue structures according to the present invention.

FIG. **7** illustrates the Control Point Spanning Tree.

FIG. **8***a* shows a graphical representation of a typical header for packets transmitted in a network such as represented in FIG. **2**.

FIG. **8***b* shows a graphical representation of a Multicast Tree Routing Field in the header represented in FIG. **8***a*.

FIG. **9** represents the List Pointer structure according to the present invention.

FIG. **10** shows an overview of the multicast mechanism as claimed in the present invention.

FIG. **11** represents the Buffer List structure of a packet supporting the multicasting routing according to the present invention.

FIG. **12** represents the Free Buffer List structure according to the present invention.

FIG. **13** represents the Processor functional structure according to the present invention.

FIG. **14** shows a general view of the packet processing in the programmable adapter according to the present invention.

FIG. **15** shows the packet routing and multicasting process according to the present invention.

FIG. **16** shows the interrupt procedure at packet level according to the present invention.

FIG. **17** shows the interrupt procedure at buffer level according to the present invention.

FIG. **18** illustrates the data flow between the Specific Purpose Processor (SPP), the Direct Access Memory (DMA) and IO1 device according to the present invention.

FIG. **19** is a flow chart illustrating the release mechanism of the memory resources in the background procedure as claimed in the present invention.

DESCRIPTION OF THE PREFERRED EMBODIMENT OF THE INVENTION

As illustrated in FIG. **2**, a typical model of communication system is made of several user networks (**212**) communicating through a high performance network (**200**) using private lines, carrier provided services, or public data networks. Each user network can be described as a set of communication processors and links (**211**) interconnecting large computers used as Enterprise Servers (**213**), user groups using workstations or personnel computers attached on LAN (Local Area Networks **214**), applications servers (**215**), PBX (Private Branch eXchange **216**) or video servers (**217**). These user networks, dispersed in different establishments, need to be interconnected through wide area transport facilities and different approaches can be used for organizing the data transfer. Some architectures involve the checking for data integrity at each network node, thus slowing down the transmission. Others are essentially looking for a high speed data transfer and to that end the transmission, routing and switching techniques within the nodes are optimized to process the flowing packets towards their final destination at the highest possible rate. The present invention belongs essentially to the latter category and more particularly to the fast packet switching network architecture detailed in the following paragraphs.

High Speed Packet Switching Networks

The general view in FIG. **2** shows a fast packet switching transmission system comprising eight nodes (**201** to **208**) each node being interconnected by means of high speed communication lines called Trunks (**209**). The access (**210**) to the high speed network by the users is realized through Access Nodes (**202** to **205**) located at the periphery. These Access Nodes comprise one or more Ports, each one pro-

viding an access point for attaching external devices supporting standard interfaces to the network and performing the conversions required to transport the users data flow across the network from and to other external devices. As example, the Access Node **202** interfaces respectively a Private Branch eXchange (PBX), an application server and a hub through three Ports and communicates through the network by means of the adjacent Transit Nodes **201**, **208** and **205**.

Switching Nodes

Each network node (**201** to **208**) includes a Routing Point where the incoming data packets are selectively routed on the outgoing Trunks towards the neighboring Transit Nodes. Such routing decisions are made according to the information contained in the header of the data packets. In addition to the basic packet routing function, the network nodes also provide ancillary services such as:

the determination of routing paths for packets originated in the node,

directory services like retrieving and updating information about network users and resources,

the maintaining of a consistent view of the physical network topology, including link utilization information, and

the reservation of resources at access points of the network.

Each Port is connected to a plurality of user processing equipments, each user equipment comprising either a source of digital data to be transmitted to another user system, or a data sink for consuming digital data received from another user system, or, typically, both. The interpretation of the users protocols, the translation of the users data into packets formatted appropriately for their transmission on the packet network (**200**) and the generation of a header to route these packets are executed by an Access Agent running in the Port. This network layer header (**800**) shown in FIG. *8a* is made of Control (**801**), Routing (**802**) and Redundancy Check (**803**) Fields.

The Control Fields (**801**) include, among other things, an encoded identification of the protocol to be used in interpreting the Routing Field.

The Routing Fields (**802**) contain all the information necessary to route the packet through the network (**200**) to the destination End Node to which it is addressed. These fields can take several formats depending on the routing mode specified

The Redundancy Check Fields (**803**) are used to check for errors in the header itself. If an error is detected, the packet is discarded.

Routing Points

FIG. **3** shows a general block diagram of a typical Routing Point (**300**) such as it can be found in the network Nodes (**201** to **208**) illustrated in FIG. **2**. A Routing Point comprises a high speed packet Switch (**302**) onto which packets arriving at the Routing Point are entered. Such packets are received:

from other nodes over high speed transmission links (**303**) via Trunk Adapters (**304**).

from users via application adapters called Ports (**301**).

Using information in the packet header, the adapters (**304**, **301**) determine which packets are to be routed by means of the Switch (**302**) towards a local user network (**307**) or towards a transmission link (**303**) leaving the Node. The adapters (**301** and **304**) include queuing circuits for queuing packets prior to or subsequent to their launch on the Switch (**302**).

The Route Controller (**305**) calculates the optimum routes through the network (**200**) so as to minimize the amount of network resources used to complete a communication path and builds the header of the packets generated in the Routing Point. The optimization criteria includes the characteristics of the connection request, the capabilities and the utilization of the Trunks in the path, the number of intermediate nodes .... All the information necessary for the routing, about the nodes and transmission links connected to the nodes, are contained in a Network Topology Database (**306**). Under steady state conditions, every Routing Point has the same view of the network. The network topology information is updated when new links are activated or new nodes added to the network. Such information is exchanged by means of control messages with all other Route Controllers to provide the necessary up-to-date information needed for route calculation (such database updates are carried on packets very similar to the data packets between end users of the network). The fact that the network topology is kept current in every node through continuous updates allows dynamic network reconfigurations without disrupting end users logical sessions

The incoming transmission links to the packet Routing Point may comprise links from external devices in the local user networks (**210**) or links (Trunks) from adjacent network nodes (**209**). In any case, the Routing Point operates in the same manner to receive each data packet and forward it on to another Routing Point as dictated by the information in the packet header. The fast packet switching network operates to enable a communication between any two end user applications without dedicating any transmission or node facilities to that communication path except for the duration of a single packet. In this way, the utilization of the communication facilities of the packet network is optimized to carry significantly more traffic than would be possible with dedicated transmission links for each communication path.

Control Point Spanning Tree

Associated with networks are several network control functions: network Spanning Tree maintenance, Topology Database, Directory, Path Selection, Bandwidth Management and Reservation, and Congestion Control. Preferably, every node has a set of the foregoing network control functions called its Control Point (CP) that the node uses to facilitate the establishment of connections between user applications.

The main purpose of the Control Point Spanning Tree is to ensure a communication and distribution mechanism for all network control functions in the nodes of a high speed network. It (logically) joins together the Control Points (**305**) if the nodes are in a (physically) connected portion of the network. As illustrated in FIG. **7** a tree is a pattern of connections with no loop, the term "spanning" means that the tree spans (connects) all of the nodes. Once formed, the Control Point Spanning Tree is the principal system used to disseminate control information such as Topology Database (**306**) updates. This mechanism is fundamental to minimize delays due to intermediate node processing.

First, an intermediate node will get each control message exactly once on the tree, and

second, the message can be forwarded along outgoing links of the tree before the intermediate node has even looked at the packet contents.

A distributed algorithm creates and maintains the Control Point Spanning Tree in presence of node and link failures and helps to minimize the impact of the increased control flows that result when the network grows.

Routing Modes

The routing within the network presents two aspects:

1. Determining what the route for a given connection shall be.

2. Actually switching the packet within a switching node.

There are many methods of determining a route through a network. For very high throughput, once the route selected, the critical item is that the switching elements must be able to route an incoming packet in a very short portion of time. Driven by the requirements to keep transit node processing at a minimum, the transport services are designed to operate on an end-to-end basis so there is no hop-by-hop error recovery or retransmission envisioned for high speed, high performance (low error) links. There is also no need for transit nodes to be aware of individual transport connections.

Data packets are routed and queued in the transit nodes according to the routing information contained in the header. Several routing modes can be used in high speed networks (refer to an *Introductory Survey* (pages 116 to 129)—GG24-3816-01 ITSC Raleigh June 1993). However, in most of the case, packets using different modes can share the same data transmission facilities.

To classify the different transport type, we can consider on one side the point-to-point transmissions on the other side the multi-pointss transmissions. As illustrated by the following examples, each routing mode has its particular indented use and includes advantages and disadvantages that complement the other modes.:

Point-to-Point Transmission

Source Routing

The Source Routing is a particular implementation of the distributed routing for connectionless networks. The The source node (or access node) is responsible for calculating the route the packet must take through the network.

Each packet includes in its routing field a list of the labels of all links through which the packet will pass as it moves across the network. The Source Routing requires no connection setup activity in intermediate nodes and supports true datagram services.

Label Swapping

This routing mode is used in connection oriented networks. Each packet sent on the link has a header which includes an arbitrary number identifying which logical connection that this packet belongs to. Label Swapping requires that connection tables be set up and maintained dynamically in each node. Due to the low packet overhead, this technique is particularly adapted to the transmission of very short packets (for example real-time voice connections).

Multi-Points Transmission

Multicast allows one entity to communicate with multiple entities. An efficient multicast mechanism provides packet delivery to a set of users without having to broadcast to all users in the network or having to "unicast" separate copies of packets to each user of a group.

Multicast Tree Routing

Multicast Tree Routing is supported by the ability of each node to recognize that a particular label represents a predefined tree and to forward the received packet along all the outbound links associated with that tree. This is the basic routing mechanism that underlies both the fundamental Control Point Spanning Tree and any individual trees that can be established, pruned, and removed dynamically in a network. The Control Point Spanning Tree greatly reduces the negative effect on performance that network control flows could have in a high speed network. Tree routing on other Multicast Trees can be fundamental to supporting, for example, multi-attached LANs, or specific applications like video conference.

Remote Access to a Multicast Tree

Remote Access to a Broadcast Tree is a straightforward combination of the two routing modes: Source Routing and Multicast Tree Routing. The routing field includes several Source Routing labels followed by a tree address label. This allows a node that is not a member of a tree to route packets to nodes that are tree members. This technique is described with more details in European patent application 93480059.0 filed May 19th 1993 entitled *Method and Apparatus for Routing Packets in Packet Transmission Networks.*

Multicast Mechanism

The Multicast Tree Routing provides a broadcast capability over a preselected tree that can connect multiple nodes. Any member of a multicast group can send a packet which will be forwarded to all other member of the group. A Multicast Tree is simply identified, in the Routing Point, by a reference in the Topology Database. Different trees can coexist in the same network. One internal use of a Multicast Tree is to join all the Control Points of a network (Control Point Spanning Tree) for maintaining in each Topology Database a true image of the network configuration. This broadcast mechanism can also be used to address subset of network users (like a closer user group).

As illustrated in FIG. 8*b*, a Multicast Tree is identified by a tree address (**804**) which is part of the routing field (**802**) of packets to be sent on the tree. Unlike the Source Routing labels which are uniquely preassigned to individual links within each node, the tree address is assigned to all links that are to be part of a tree at the time that the tree is set up. The tree address must be unique to one and only one tree across all nodes over which the tree passes. When a packet addressed to a Multicast Tree arrives at a node over a link, any links in that node that are configured for that tree address will forward this packet. In this manner, copies of the packet flow across the network, along each link configured as a branch of the tree, with one and only one copy arriving at each node that is part of the tree. A Hop Countdown Field (**805**) is included in the Multicast Tree Routing Field (**802**) which is decremented at each retransmission of the packet. When the hop countdown is equal to zero, no further retransmission is permitted and an error condition is assumed. The routing field is terminated by and End Of Field (EOF) flag (**806**).

Any node can be simultaneously be a member of a plurality of different trees and hence the tree addresses assigned to overlapping multicast trees must be unique. Nodes can be added or deleted from a Multicast Tree simply by adding or removing the tree address from the links leading to the node to be added or deleted. A more detailed description is disclosed in European patent application 93480060.8 filed May 19th 1993 entitled *Multicast Com-*

munication Tree Creation and Control Method and Apparatus.

Ports and Trunk Adapters

Adapters Function

Ports are located at the boundary of the high speed network. They allow terminal equipments to exchange information through the high speed network without the need for knowing the specific high speed protocol used. The main function of the Ports are:

receiving foreign protocol data units from an external resource and forwarding them as high speed packets over the network to a target Port, and

converting high speed packets back to foreign protocol data units and sending them to the target resource,

controlling the bandwidth.

Note: the source and target Ports may be located in the same node.

Trunks are the links between the high speed network nodes. They carry high speed packets. Each Trunk manage its link bandwidth and link status. A critical task of the Trunks is the management of traffic priorities and allocation of internal buffers to reduce delay and congestion.

In addition, there is a special type of adapter called Route Controller Adapter (305) which:

communicates with the other adapters (301, 304) through the Switch (302),

implements the centralized functions of the Route Controller such as the topology, the path selection . . . ,

establishes, maintains and cancels end-to-end high speed connections.

Adapters Architecture

Several techniques for designing said Ports, Trunk and Route Controller Adapters, to obtain more or less flexible and efficient transmission systems. Most of the adapters today are built around a specialized hardware depending on the function and protocol of the connected links.

The present invention, to satisfy the previously enumerated connectivity and flexibility requirements, provides a software solution based on a common hardware structure. Port and Trunk Adapters present the same architecture and their functional differentiation is realized through a specific programming. However, even using the most efficient general purpose microprocessor today available on the market, the experience shows that it is very difficult to reach the desired level of performance in term of number of switched packet per second. This is the reason why the control of each adapter has been shared between two processors: a Specific Purpose Processors (SPP, 406, 412), optimized for the packet switching and a General Purpose Processor (GPP, 409), the first dealing with the packet to be switched, the critical processing in term of performance, and the second with the adapter management.

As shown in FIG. 4, each adapter (400) comprises the following logic components:

1. A General Purpose Processor (GPP, 409) whose programming depends of the selected Port or Trunk Adapter function. The GPP implements the adapter control operations.

2. A Receive Adapter (401) for implementing three functions:

the checking of the high speed packets header.

the traffic discrimination according to the routing mode specified in the header of every incoming packet,

the routing of the incoming packets through the Switch (403) with the appropriate header.

The Receive Adapter includes:

a. a Line Receiver (407) for handling the data movements between a Line Interface (415) and a Receive Buffer Memory (RBM, 405).

b. a Receive Buffer Memory (RBM, 405) to temporarily store users data packets.

c. a Receive Specific Purpose Processor (RSPP, 406) based on a specialized microprocessor comprising a Local Memory (LM, 408). The RSPP handles the received steady state packet flow and forwards the control packets to the General Purpose Processor (409).

d. a Local Memory (LM, 408) used by the RSPP (406) as work space.

e. a Switch Transmitter Adapter (404) for

handling the data flow transferred from the buffer Memory (RBM, 405) under the control of the Receive Specific Purpose Processor (406)

segmenting this flow in fixed length cells and,

generating an appropriate switch routing header

3. a Transmit Adapter (402) for implementing the following functions:

the reception of the data flow from the Switch (403),

the checking of the cells header.

the reassembly in packets (Port),

the Trunk functions (Trunk adapter),

the routing.

The Transmit Adapter includes:

a. a Switch Receiver (410) for handling the flow coming from the Switch (403) and transferring it to the Buffer Memory for reassembly.

b. a Transmit Specific Purpose Processor (XSPP, 412) similar to the Receive Specific Purpose Processor (406). The XSPP handles the steady state data and forwards the control flow to the General Purpose Processor (GPP, 409).

c. a Line Transmitter Adapter (413) for handling the data movements between the Buffer Memory (411) and the Line Interface (415).

The adapters are connected on one side on the packet Switch and on the other side on the Line Interfaces:

The Line Interfaces (415) are used to adapt the Port and Trunk Adapter physical interfaces to the appropriate media.

The packet Switch (302, 403) allows the Route Controller (305) and the different Ports (301), Trunk Adapters (304) to communicate.

Data Flow Control

The receive and transmit data flows in the Trunk Adapters are represented in FIG. 5. In a proprietary high speed network with packets of variable lengths, the receive process involves the steps of:

1. Line Receiver, Buffer Memory, Specific Purpose Processor (501) system

a. receiving the packets from the line,

b. checking the packets header and in case of error discarding the packets,

c. processing the information contained in the packets header according the routing mode,

d. routing the control messages towards the General Purpose Processor (GPP, 502)

e. encapsulating the packets with a specific switch header in function of the destination adapter,

f. forwarding the packets and the GPP (502) control messages to the Switch Transmitter Adapter (504),

2. Switch Transmitter Adapter (504)

a. segmenting the packets in cells of fixed length adapted to the Switch (503),

b. generating an error check field to ensure the integrity, of the switch header during the transmission of the cells over the Switch (503).

The transmit process, as for it, comprises the steps of:

1. Switch Receiver Adapter (505)

a. receiving the cells from the Switch (503),

b. checking the switch header and, in case of error, discarding the cell,

2. Line Receiver, Buffer Memory, Specific Purpose Processor (506) system

a. reassembling the data packets,

b. forwarding the control packets to the General Purpose Processor (502),

c. encapsulating the packets with a routing header,

d. receiving control packets from the GPP (502),

e. queueing data and control packets in the appropriate queues,

f. handling the outgoing packets with priority given to real time data (and then to non real time data).

It is possible to design the adapters to work either in a a proprietary environment with packets of variable length, or in a standard mode such as ATM (Asynchronous Transmission Mode) with short cells of fixed length, or in both where appropriate. In this last case, for performance purpose, cells routed on the switch are identical or similar to these defined in the ATM protocol with as result:

the elimination of the packets segmentation and reassembly steps in the Switch Transmitter (508) and Receiver Adapters (509),

a simplification of the switch header processing in the Specific Purpose Processor (507, 510).

Adapter Functional Structure

The present invention deals with the relationships between the Line Receiver/Transmitter, the Buffer Memory, the Specific Purpose Processor, and the Switch Adapter and in particular with the handling of the data flow in a way to optimize the throughput and the processing time inside the adapters. More specifically, the invention relates to a very high performance system for queuing, dequeueing and distributing data packets on external links.

The communication adapters are based on the following principles:

the Specific Purpose Processor is designed to minimize the number of operations necessary to manage the steady state data flow.

data packets and control data are managed separately in two distinct memories respectively the Buffer Memory and the Local Memory.

the data packets buffering, the queueing, dequeuing, the routing and multicasting mechanisms are identical for all Ports—Trunk Receive and Transmit adapters.

According to these considerations, the following conventions will be used to describe the invention:

The device which reads in the Buffer Memory and the device which writes into the Buffer Memory are designated respectively by IO1 and IO2. That means:

in the receive side of the adapter, IO1=Switch Transmitter and IO2=Line Receiver

in the transmit side of the adapter, IO1=Line Transmitter and IO2=Switch Receiver.

In the same way:

an input data stream goes from the switch or external line (IO2) to the Buffer Memory.

an output data stream goes from the Buffer Memory to the switch or external line (IO1).

Furthermore:

The meaning of "packet" is application dependent. It may be applied, for example, to an SDLC-frame from a Port, to a proprietary packet format from a Trunk, or to a cell received from an ATM Trunk. The term "packet" that will be used in the following paragraphs will not refer to a precise data unit format.

Data Structures

Buffers, Packets, Queues Structures

Data packets are stored in the Buffer Memory (BM) while control data are managed directly by the Specific Purpose Processor (SPP) in the Local Memory (LM). The basic unit of memory that can be allocated to an input (Line/Switch Receiver (IO2)) or output device (Line/Switch Transmitter (IO1)) is a buffer of fixed length. As illustrated in FIG. 6, each of these buffers is represented in the Local Memory by a pointer called Buffer Pointer (B_PTR). A pointer is a generic term to identify a logical data structure (buffer, packet, queue . . .) stored in the Buffer Memory. The storage of a packet requires one or several buffers. These buffers are stacked together using a list of pointers (B_LIST) which is itself represented by a Packet Pointer (P_PTR). A list of Packet Pointers (P_LIST), identified by a Queue Pointer (Q_PTR), designates a queue of several packets.

List Prefix

Each list, representing a specific packet or a queue structure, is preceded by a Prefix used for storing any type of information related to the data the structure contains. In Buffer Lists, the Prefix contains information related to the routing of the packet:

the packet header

the date of the packet reception

the packet length

All the processor operations on the packet header are realized in the the Local Memory (LM) without having to access to the data stored in the Buffer Memory (BM). Furthermore, when the processor (SPP) is working on the Local Memory, the DMA operations on the Buffer Memory are not disrupted. The result is a more efficient routing process and memory management. As illustrated in FIG. 11, the support of the multicasting mechanism is realized by means of specific fields in the Buffer List Prefix of a each packet:

a Multicast Counter (MCC)

The counter is initialized according to the number of outputs. The initial counter value is one or superior to one depending on if the packet is intended to be transmitted to a single (MCC=1: Point-to-point Routing Mode) or to multiple destinations (MCC>1: Multi-points Routing

Mode). The counter value is decremented each time the packet to multicast is transmitted to an output. When the counter value is equal to zero, the routing process is terminated and the resources attached to the packet can be released.

a copy of the Packet Pointer (P_PTR)

The original Packet Pointer, with the initial HEAD (identification of the first Buffer Pointer in the list) and TAIL (identification of the next Buffer Pointer in the list) values, is saved in the Buffer List Prefix to allow, at the end of the transmission, the complete release of all Buffers attached to the packet.

Packet Segmentation

To facilitate the memory management, the lists used for packets and queues are of fixed length. Packets which are bigger than the buffer list can contain, are segmented. This method allows the lists (B_LIST) not to be sized at the maximum packet length.

Buffer Pointers

Buffers need not to be full and the data may start and end at any place. For example, it is possible to reserve the first bytes of a buffer to include a header a posteriori. A Status Field (SF) is used in the last Buffer Pointer of a list to designate an End Of Packet (EOP). The general format of a Buffer Pointers is described in European patent application 93480087.1 (IBM docket FR993027) entitled *Programmable High Performance Data Communication Adapter for Packet Transmission Networks*

List Pointers

Referring to FIG. 9, the List Pointer format consists of three fields:

the List IDentifier (LID): identification of the list,

the HEAD: identification of the first pointer of the list,

the TAIL: identification of the next pointer to attach to the list.

The queuing process comprises the steps of:

testing if list is full (if INCREMENTED(TAIL)=HEAD).

storing the pointer at the address identified by the TAIL field in the pointer list identified by the LID field,

incrementing the TAIL field of the List Pointer,

The dequeueing process comprises the steps of:

testing if the list is not empty (if HEAD not equal to TAIL).

reading the pointer at the address identified by the HEAD field in the pointer list identified by the LID field,

incrementing the HEAD field of the List Pointer when list is not empty.

This queueing and dequeueing method is described with more details in European patent application 93480087.1 (IBM docket FR993027).

Free Buffer List (FBL)

The management of the Buffer Memory is realized by means of a specific list called Free Buffer List (FBL). The FBL comprises the totality of the Buffer Pointers and its role is to provide a status of the memory occupation (FIG. 12) using the HEAD and TAIL fields of the Free Buffer List Pointer (P_FBL):

T: total number of buffers in the Buffer Memory.

HEAD: identification of the first free buffer of the list. Each time a new buffer is filled, the HEAD field is incremented.

TAIL: identification of the next free buffer of the list. Each time a new buffer is released, the TAIL field is incremented.

HEAD=TAIL: the Buffer Memory is full.

Incremented TAIL=HEAD: the Buffer Memory is empty.

The Free Buffer List (FBL) is created at initiation time and is permanent contrary to other lists which are created dynamically (Buffer, Packet or Queue Lists).

Note: in general, when a lack of resources is detected (Free Buffer List empty), then the packet which cannot be stored in the Buffer Memory is discarded.

Free Packet List (FPL)

Buffer Lists are of fixed length (fixed number of elements). The management of these lists in the Local Memory (LM) is realized by means of a specific list called Free Packet List (FPL). The FPL is created at initiation time and is permanent like the Free Buffer List. It comprises all the Buffer List structures that will be dynamically created or released during the packet receive and transmission process.

It is important to note that, in the present invention, pointers (Buffer, Packet or Queue pointers) are simply stacked in predefined lists (FIG. 6). The pointers are not chained that means that pointers do not contain the address of the next pointer in the list and they are not aware of the location of this next pointer. This list structure is designed to optimize in term of performance the buffer manipulation in very high speed adapters but not the memory resources.

Specific Purpose Processor Structure

The Specific Purpose Processor functional structure is illustrated in FIG. 13.

Processor Parallel Processing

The Specific Purpose Processor is designed to execute up to three operations in parallel:

1. ALU (Arithmetical and Logical Unit) operations on registers

2. Memory operations

3. Sequence operations

The parallelism requires to distinguish instructions from operations:

Instruction: it is the content of the code word. In term of assembler language, the instruction corresponds to one line of code. All instructions are executed in one processor cycle

Operation: an instruction may be composed of one or more operations which are executed simultaneously.

Memory Space

The SPP memory space is divided in three blocks:

the Instruction Store (130),

The Local Memory (LM, 131), which is the code working space,

The Buffer Memory (BM, 132) which is the repository for data packets when they pass through the adapter.

They all operate in parallel, the Instruction Store (130) under control of the Sequencer (133), the Local Memory under control of the processor code and the Buffer Memory (132) under the control of the Direct Access Memory (DMA, 134)).

Registers

The registers are divided in two categories:

1. the General Purpose Registers (GPR)

These registers are located in the Register File (RF, 135) and are available as instruction operands.

2. the Control Registers (CR)

The CR's are hardware registers which are used in specific functions and are available also as instruction operands. However, there are less degree of freedom in their use, as compared with the GPR's. In particular, two of these control registers (136) are located in the Direct Access Memory (DMA, 134).

CR1=D_PTR1 (DMA Pointer IO1)

CR2=D_PTR2 (DMA Pointer IO2)

DMA Pointers 1 and 2 are associated to input/output IO1 (137) and IO2 (138) and they both contain the current Buffer Pointer (B_PTR).

Memory Address Generator (MAG, 139)

In all load or store operations, on the Local or on the Buffer Memory, the physical address is reconstituted from the different fields of Buffer or List Pointer, used as operand. For performance reason, this operation is realized by a specialized hardware component called Memory Address Generator (MAG,139).

Direct Memory Access Controller (DMA, 134))

The use of a Direct Memory Access Controller (DMA, 134) jointly with a processor is well-known in the state of the art. Its role is to quickly move the data packets between the IO devices (137, 138) and the Buffer Memory (132) without the processor (SPP) intervention. The DMA module consists of two independent programmable channels. The IO devices present their service requests (SR1, SR2) to the DMA which controls the access to the Buffer Memory (132). The processor intervention is needed only at buffer and packet boundaries. The data streams between the two IO devices and the Buffer Memory is processed in parallel with the code. Up two IO operations can be multiplexed on the BMIO bus; one with IO1 and the other with IO2. For that the DMA manages the two DMA Pointers (D_PTR1 and D_PTR2) which are nothing else than Buffer Pointers.

Input Output Subsystems

The Specific Purpose Processor (SPP) is considered as the "master" and it establishes the connections.

The IO devices and the Buffer Memory are controlled either directly by the processor code, or via the DMA.

The code intervention can be forced by the IO devices via the Interrupt mechanism in case of buffer or packet boundary.

Data Reception and Transmission

Various processing can be made on Buffer and List Pointers:

Incrementing a Buffer Pointer,

Closing a buffer,

Accessing a List Prefix,

Attaching an element to a list,

Detaching an element from a list.

Operations on Pointers

Some operations on pointers are performed by the processor code, others by the Direct Memory Access (DMA).

The writing and reading of the Buffer Pointers is exclusively the fact of the DMA (134).

The writing in the Buffer Memory: At the reception of a service request (SR2) from an IO2, the DMA has access to the Buffer Memory (132) by means of the address contained in the Pointer 2 (D_PTR2, 136). The DMA Pointer 2 is provided by the processor (SPP) and is nothing else than a Buffer Pointer (B_PTR). The DMA orders simultaneously the IO2 (138) to present a data element on the BMIO bus and the Buffer Memory (132) to write this data element in the buffer identified by the DMA Pointer. Data are filled into the buffer starting an address chosen by the code and ending at the bottom of the buffer except the last buffer (of a packet) where data may end at any place. When the buffer is full, the DMA demands from the processor a new Buffer Pointer through an interrupt mechanism (IO2_EOB routine). A similar procedure is used when the IO2 detects the end of a packet (IO2_EOP routine)).

The Reading in the Buffer Memory: At the reception of a service request (SR1) from an IO1, the DMA addresses the Buffer Memory (132) by means of the DMA Pointer 1 (D_PTR1, 136). The DMA Pointer 1 is provided by the processor (SPP). The DMA orders simultaneously the Buffer Memory (132) to present on the BMIO bus a data element in the buffer identified by the DMA Pointer and the IO1 device (138) to read this data element. When the buffer is empty, the DMA demands a new Buffer Pointer from the processor through an interrupt mechanism (IO1_EOB routine). A similar procedure is used when the DMA detects a end of packet (IO1_EOP routine). After the data transfer, the Buffer Pointer is released in the Free Buffer List, Packet List and Queue List are updated accordingly.

Packet and Queue Pointers are managed by the processor code:

the Code intervention is forced by the IO devices and DMA via the Interrupt mechanism in case of buffer or packet boundary.

the buffer and packet queueing and dequeueing mechanisms are executed under the control of the processor in the Local Memory.

Interrupts

The Interrupt mechanism is the way real time is supported by the Specific Purpose Processor (SPP). An Interrupt is a break, due to particular events, in the normal sequence of the processor code. The events which can cause this Interrupt are the service requests from the IO devices associated with specific conditions such as end of buffer, end of packet . . . . At each specific Interrupt corresponds a specific routine which cannot be interrupted by another routine. It is very important to have interrupt routines as short as possible to avoid overrun/underrun problems.

1. The following Interrupts are used by the Line and Switch Transmitters (IO1).

IO1_EOB:

Condition: when serving an output IO1, the DMA has emptied a buffer (which is not the last buffer of the packet/segment—the Buffer Pointer is not flagged EOS or EOP) the

DMA Pointer (D_PTR1) raises a IO1_EOB interrupt which triggers a IO1_EOB routine.

Routine: this routine

releases the pointer of the just emptied buffer in the Free Buffer List. A new pointer is dequeued from the Output Buffer List (OB_LIST) and passed to the DMA Pointer (D_PTR1).

IO1_EOP:

Condition: when serving an output IO1, the DMA has emptied a packet (the last buffer of a packet which pointer contains the EOP flag ON), the DMA Pointer (D_PTR1) raises a IO1_EOP interrupt which triggers a IO1_EOP routine.

Routine: the routine:

releases the current and last Buffer Pointer of the Output Buffer List (OB_LIST) in the Free Buffer List (FBL).

releases the current Packet Pointer in the Free Packet List (FPL)

detaches the current Output Packet Pointer (OP_PTR) from the Output Packet List (OP_LIST).

dequeues the next packet and its pointer from the Output Packet List

2. These Interrupts are used by the Line and the Switch Receiver (IO2).

$IO2_{13}$ EOB:

Condition: when serving an input IO2, the DMA has detected a Buffer Full condition (That buffer is not the last buffer of a packet), the DMA Pointer (D_PTR2) raises a IO2_EOB interrupt which triggers a IO2_EOB routine.

Routine this routine:

stores the pointer of the just filled up buffer in a pre-allocated Input Buffer List (IB_LIST) where all buffers of the same packet are stacked.

updates the Input Buffer List Prefix area.

provides the DMA Pointer (D_PTR2) with a new Buffer Pointer for continuing the reception of the data of the same packet. Free Buffer Pointers are managed by the Free Buffer List (FBL).

when for the same packet, the Buffer List is full, the segmentation takes place.

IO2_EOP:

Condition: when serving an input IO2, the DMA has completed the reception of a packet (the last buffer of the packet which pointer is flagged EOP by the hardware), the DMA Pointer raises a IO2_EOP interrupt which triggers a IO2_EOP routine. The code intervention is required to provide a new Buffer Pointer for the reception of a new packet.

Routine: this routine:

stores the pointer of the last filled buffer in a pre-allocated Input Buffer List (IB_LIST) where all buffers of the same packet are automatically stacked. The last Buffer Pointer of a packet is flagged EOP by the DMA Pointer (D_PTR2).

updates the Input Buffer List Prefix area.

the Packet Pointer of the Input Buffer List is queued in the Input Packet List (IP_LIST).

provides the DMA Pointer (D_PTR2) with a new Buffer Pointer (B_PTR) for the reception of the next packet.

Point-to-Point and Multi-Point Routing

Multicasting means a packet should be sent to at least two destinations:

to different Transmit Adapters (via the Switch) in a Receive Adapter and when needed to the General Purpose Processor (GPP).

to output Trunks or Ports in a Transmit Adapter and when needed to the General Purpose Processor (GPP).

The general principles of the point-to-point and multi-points routing according to the present invention can be summarized as follows:

Data packets are never copied, only Packet Pointers are copied for each destination:

Space in Buffer Memory is saved.

The number of instructions required for the background (delayed) process is significantly reduced improving the packet throughput (number of packets per seconds that the adapter is able to transmit).

The routing is independent of the packets length.

No overhead generated by the multicasting mechanism in the interrupt (real time) procedures.

The underrun/overrun problems on the output device IO1 are reduced.

The efficiency of the adapter in term data throughput (bits per second) is significantly improved.

Each output is processed independently by means of interrupt (real time) routines:

Lines are managed in real time.

Lines of different speed or protocol can be supported in parallel.

The release of the resources is entirely realized on a no priority mode in the background (delayed) process.

Output Queues

Each destination has its own Output Queue where the packets waiting for transmission are stacked. The number of queues is limited to the number of destinations:

In the Receive Adapter (**401**), there is one Output Queue per adapter on the Switch (**403**) plus one specific Output Queue dedicated to the General Purpose Processor (GPP, **409**).

Likewise in the Transmit Adapter (**402**), there is one Output Queue per Trunk or line in output of the Line Interface (**416**) plus one specific Output Queue dedicated to the General Purpose Processor (GPP, **409**).

The Output Queues are located in the Local Memory (**131**). They are managed by the Specific Purpose Processor (SPP) by means of the background and interrupt (EOB, EOP) procedures detailed in flow charts of FIGS. **15**, **16** and **17**.

Output Queue Table

As shown in FIG. **10**, during the transmission of the data packets, the state of each Output Queue is stored in a specific table called Output Queue Table (**100**). Each Output Queue is represented by one record comprising:

the Queue Pointer (**104**)(Q_PTR) identifying the queue.

the Current Packet Pointer (**105**) (cur P_PTR) identifying, when the Output Queue is ACTIVE, the packet to transmit.

the Current Buffer Pointer (**106**) (cur B_PTR) identifying, when the Output Queue is ACTIVE, the buffer to transmit.

the state (**107**) (ST) of the Output Queue:

ACTIVE state: a START TRANSMISSION has been sent to the IO1. The packet identified by the Current Packet Pointer (cur P_PTR) is transmitted from the Buffer Memory to the selected output under the control of the DMA.

NON ACTIVE state: the transmission of the previous packet is terminated and the Output Queue is empty.

Packet Processing

FIG. 14 gives a general view of the packet processing mechanism in a programmable high speed adapter. The receive, routing and transmission process involves the steps of:

(140) Reassembling each packet received in an Input Buffer List and initializing the Buffer List Prefix with the routing information. Packets in error are discarded.

(141) Creating for each packet a specific Buffer List identified by a Packet Pointer (IP1 . . . Ipm).

(142) Processing the packets according to their routing mode with a local access to a Routing Table (148).

(143) Queueing the Packet Pointers in the Output Queues (OQ1 . . . OQN) corresponding to the destination of the packets (145). Packets intended for the GPP (146) are stacked in a specific Output Queue (144)

(149) Triggering an interrupt routine each time a new buffer is requested. Interrupts are triggered asynchronously by the DMA on the occurrence of specific events: End Of Buffer (EOB), End Of Packet (EOP).

(1410) Transmitting the data from the Buffer Memory (BM,147) to the output device (IO1) under the control of the DMA.

(1411) Scanning the outputs (145).

Background Procedure

FIG. 15 is a flow chart illustrating the the packet routing and multicasting procedure represented in FIG. 14 (142):

(150) The processor waits for the next reassembled packet.

(151) As soon a packet is ready to be processed, its Packet Pointer (P_PTR) is retrieved. The Buffer List Prefix identified by said Packet Pointer (P_PTR) is updated. FIG. 10 details the different elements of the routing mechanism:

(102) The determination of the different destinations is given through a Routing Table (156) accessed by the Routing Field (802) of the packet header. In case of multi-points routing, Multicast Trees are predefined in each nodes at the connection set-up. Routing Tables (156) are maintained dynamically. That means that when a new connection is established or an old one is terminated the tables are updated (the database of network topology can of course be maintained quite separately). The Multicast Counter (MCC) in the Buffer List Prefix is set to its initial value according to the number of destinations given in the Routing Table (156). This value is equal to one for a point-to-point routing and superior to one for a multi-points routing mode.

(101) The Packet Pointer (P_PTR) (with the original HEAD and TAIL values) is saved in the Buffer List Prefix for a further release of the Buffers in the Buffer Memory.

(152) The Packet Pointer (P_PTR) corresponding to the packet to route or multicast is queued (the TAIL value of the Queue Pointer is incremented) in the first selected Output Queue (103).

(153) If the Output Queue is in an ACTIVE state, the process start again with with the next selected Output Queue (if any).

(154) If the Output Queue is in an NON ACTIVE state (ST), the first Packet Pointer of the Output Queue is dequeued (the HEAD value of the Queue Pointer is incremented) and saved as Current Packet Pointer (cur P_PTR) in the Output Queue Table. Likewise, the first Buffer Pointer (B_PTR) of the corresponding Buffer List is dequeued (the HEAD value of the Packet Pointer is incremented) and saved as Current Buffer Pointer (cur B_PTR) in this same Output Queue Table. Then, the Specific Purpose Processor (SPP) sends to the IO1 a request to start the transmission of the Current Packet (START TRANSMISSION). The Output Queue is set in an ACTIVE state (ST) in the Output Queue Table. The process goes on with the next selected Output Queue (if any).

(155) When all the selected outputs have been successively processed the packet routing and multicasting procedure releases all resources attached to the packets and buffers already transmitted (157). It is important to note that this release takes place in the background procedure without overloading the interrupt procedures.

(150) Once the release terminated the procedure returns to its initial waiting state.

Interrupt Procedures

1. Packet Level (IO1_EOP)

FIG. 16 is a flow chart illustrating the interrupt procedure at the packet level as represented in FIG. 14 (149). On the occurrence of an End Of Packet interrupt (EOP):

(160) The Current Packet Pointer (cur P_PTR) corresponding to the last Output Queue processed (OQn) is extracted from the Output Queue Table.

(161) The Current Packet Pointer is queued in a Release Queue for a further release of the Packet Pointer in the Free Packet List and of the Buffer Pointers in the Free Buffer List. This release process is entirely realized in the background procedure (157) to reduce to a minimum the number of instructions in the interrupt procedures.

(162) The contents of the Output Queue (OQn) is tested:

(165) If Output Queue (OQn) is empty, then it is set in a NON ACTIVE state (ST) in the Output Queue Table.

(163) If Output Queue (OQn) is not empty, then the next Packet Pointer is dequeued (the HEAD value in the Queue Pointer is incremented) (166). The first Buffer Pointer of the Buffer List is dequeued (the HEAD value of the Packet Pointer is incremented) and saved (Current Buffer Pointer) in the Output Queue Table with the Packet Pointer (Current Packet Pointer). Once the Current Buffer Pointer (cur B_PTR) and the Current Packet Pointer (cur P_PTR) are identified, the Specific Purpose Processor (SPP) sends to the IO1 a request to start the transmission of the Current Packet (START TRANSMISSION).

2. Buffer Level (IO1_EOB)

FIG. 17 is a flow chart illustrating the interrupt procedure at the buffer level as represented in FIG. 14 (149). On the occurrence of an End Of Buffer interrupt (EOB):

(170) The Current Packet Pointer (cur P_PTR) corresponding to the last Output Queue processed (OQn) is extracted from the Output Queue Table.

(171) The next Buffer Pointer (B_PTR) in the Buffer List is dequeued (the HEAD value of the Packet Pointer is incremented) This Buffer Pointer is first, set into the DMA Pointer (D_PTR1) attached to the IO1 device and second, saved as Current Buffer Pointer (cur B_PTR) in the Output Queue Table. The Packet Pointer is also saved in the Output Queue Table as Current Packet Pointer.

Release of Resources

FIG. 19 is a flow chart illustrating the release of the resources attached to the packets after their transmission as represented in FIG. 15 (157). This process is entirely located in the background procedure to minimize the number of instructions during the interrupts.

(190) The contents of the Release Queue located in the Local Memory (131) is tested:

If the Release Queue is empty, the release process is terminated and the background procedure can return in its initial waiting state (150).

If the Release Queue is not empty then:

(191) A Packet Pointer is dequeued from the Release Queue (the HEAD value of the Release Queue Pointer is incremented).

(191) The Multicast Counter (MCC) is retrieved from the Buffer List Prefix.

(191) The Multicast Counter (MCC) is decremented by one and tested (192):

(193) If the Multicast Counter (MCC) value is equal to zero, then the transmission of the packet is terminated. All the resources attached to this packet must be released. The original Packet Pointer (P_PTR) (with the initial HEAD and TAIL values) previously saved is retrieved from the Buffer List Prefix. The Packet Pointer and Buffers Pointers are released respectively from the Free Packet List (FPL) and from the Free Buffer List (FBL).

If the Multicast Counter (MCC) value is not equal to zero, then the multicast process is not terminated.

(190) The contents of the Release Queue is tested one more time:

If the Release Queue is not empty, the procedure goes on with a new Packet Pointer (191).

If the Release Queue is empty, the release process is terminated and the background procedure can return in its initial waiting state (150).

Data Transmission

FIG. 18 illustrates the data flow between the Specific Purpose Processor (SPP), the Direct Access Memory (DMA) and IO1 device. First, the background procedure already described in FIG. 15 is in a waiting state. As soon as a packet is ready to be transmitted, the Packet Pointer is copied in the appropriate Output Queue n. This Output Queue n is set in an ACTIVE state and the SPP informs the IO1 device that it is ready to start the transmission on the output n (START TRANSMISSION n). At the reception of this message, the IO1 responds to the DMA with a Service Request (SR1 n) to request a first Buffer Pointer. the DMA triggers an End Of Buffer interrupt (IO1_EOB n as shown in FIG. 17) to extract from the Output Queue Table the Current Buffer Pointer (cur B_PTR n) of the Output Queue n. The DMA orders simultaneously the Buffer Memory (BM) to present the buffer identified by said Current Buffer Pointer on the BMIO and the IO1 to read this buffer (DS1 n Data Service message). The process goes on with the next buffer until the end of the packet. At this time the DMA raises an End Of Packet interrupt (IO1_EOP n as shown in FIG. 16) and informs the IO1 device (EOP1 n).

It is important to notice that all Output Queues are processed independently and in parallel under control of the IO1 device according to the different protocols and speeds used on the output lines.

We claim:

1. A line adapter for a packet switching node in a communication network, including a programmable processing means (SPP) for receiving and transmitting data packets of fixed or variable length to one or more outputs, said line adapters comprising:

a first storing means for buffering each data packet in one or more buffers;

means for identifying said buffers,

a second storing means including means for queuing said buffer identifiers in buffer lists, each buffer list identifying a data packet;

means for identifying said buffer lists;

means for queuing, in said second storing means, buffer list identifiers in packet lists, each packet list identifying a queue;

means for identifying said packet lists;

means for processing a routing header of each data packet;

means for associating with each output an output queue for stacking the packet list identifiers of the data packets to transmit on said outputs;

means for routing the data packets to one or a plurality of outputs;

means for processing said routing header further comprising means for determining and selecting the output queue corresponding to the destination of each data packet;

said means for routing further comprising:

means for copying the buffer list identifier of the data packet to transmit in the output queue corresponding to said selected output;

means for handling each data request for said outputs in real time;

means for releasing said buffers in said first storing means, and said buffer identifiers and said buffer list identifiers from said second storing means;

said means for handling data requests further comprising means for handling independently the output queues, which further includes:

means for dequeuing in parallel the packet identifiers requested by the outputs; and

means for dequeuing the buffer identifiers stacked in the buffer list identified by said packet identifiers.

2. The line adapter according to claim 1 wherein:

said first storing means includes means for writing and reading said data packets in buffers of fixed length independently of said programmable processing means; and

said second storing means includes means for separately storing said buffer lists and said packet lists under control of said programmable processing means.

3. The line adapter according to claim 1 wherein said programmable processing means (SPP) comprises:

a background procedure for processing the data packets,

a plurality of real time procedures triggered under control of each output to request the contents of a new buffer, said real time procedures being independent of the routing mode used.

4. The line adapter according to claim 3 wherein said means for releasing in said means for routing is executed in said background procedure.

5. The line adapter according to claim 1 wherein:

said means for identifying said buffers includes buffer pointers (B_PTR) identifying said buffers and stacked in one or more buffer lists (B_LIST);

said means for identifying said buffer lists includes packet pointers (P_PTR) identifying said buffer lists (B_LIST) and stacked in one or more packet lists (P_LIST); and

said means for identifying said packet lists include queue pointers (Q_PTR) identifying said packet lists (P_LIST) and stacked in one or more queue lists (Q_LIST); and

wherein each buffer list, packet list and queue list includes a prefix for storing information related to the data contained in said each buffer list, packet list and queue list.

6. The line adapter according to claim 5 wherein said buffer list prefix includes control and routing information contained in the data packet header.

7. The line adapter according to claim 6 wherein said buffer list prefix further includes a multicast counter (MCC COUNTER) and a packet pointer (P_PTR).

8. The line adapter according to claim 7 wherein a status of each output queue is stored in an output queue table located in said second storing means, said status including:

the output queue pointer (Q_PTR),

a current packet pointer (cur P_PTR) for the last packet dequeued from the output queue;

a current buffer pointer (cur B_PTR) for the last buffer dequeued from the current packet pointer; and

a state of the output queue depending on if the output queue is active, an output queue being active during the time said data packet identified by said current packet pointer is transmitted to said selected output.

9. The line adapter according to claim 8 wherein said background procedure includes:

means for detecting a new buffer list;

means for reading the routing information in the buffer list prefix;

means for initializing the multicast counter in the buffer list prefix with the number of outputs towards which the data packet must be transmitted;

means for copying the packet pointer of the buffer list in its own prefix;

means for selecting an output queue according to the destination of the data packet, queuing in said output queue the packet pointer and testing the state of said output queue, wherein:

if the state is active a new output queue is selected according to the other destinations of the data packet;

if the state is not active:

a packet pointer is dequeued from said output queue and saved as the current packet pointer in said output queue table;

a buffer pointer is dequeued from the buffer list identified by said current packet pointer and saved as the current buffer pointer in said output queue table;

a message is sent to the output;

the output queue is set in an active state; and

a new output queue is selected according to the other destinations of the data packet;

when said packet pointer has been queued in all selected output queues the background process returns to its waiting state.

10. The line adapter according to claim 9 wherein said plurality of real time procedures comprise:

a first real time procedure triggered by the outputs on request for a new buffer including:

means for getting the packet pointer corresponding to the last output queue processed in the output queue table;

means for dequeing the next buffer pointer in the buffer list identified by said packet pointer;

means for saving said packet pointer into the output queue table as the current packet pointer;

means for saving said buffer pointer into the output queue table as the current buffer pointer;

a second real time procedure triggered by the outputs on request for a new packet including:

means for getting the packet pointer corresponding to the last output queue processed in the output queue table;

means for queuing said packet pointer in a release queue for a delayed release;

means for testing if the output queue is empty or not, wherein:

if said output queue is empty, said output queue is set in a non-active state;

if said output queue is not empty:

a next packet pointer is dequeued from said output queue;

said next packet pointer is saved as the current packet pointer in the output queue table;

a first buffer pointer of the buffer list identified by said packet pointer is saved as the current buffer pointer in the output queue table; and

a message is sent to the output associated with said output queue.

11. The line adapter according to claim 1 wherein the management of the buffers in said first storing means is realized by means of a first permanent list (free buffer list) containing all of the buffer pointers.

12. The line adapter according to claim 1 wherein the management of the buffer lists in said second storing means is realized by means of a second permanent list (free packet list) containing all of the packet pointers.

13. The line adapter according to claim 1 wherein said releasing means in said background procedure comprises:

means for testing the contents of the release queue:

if the release queue is empty, then terminating the release;

if the release queue is not empty:

dequeuing a packet pointer from the release queue;

decrementing the buffer list prefix counter by one;

testing if the multicast counter value is equal to zero,

if the multicast counter value is not zero, then testing the contents of the release queue:

if the release queue is empty, then terminating the release;

if the release queue is not empty, then dequeuing the next packet pointer from the release queue;

if the multicast counter value is zero:

retrieving the original packet pointer from the buffer list prefix identified by said packet pointer;

releasing the buffer list and buffers identified by said packet pointer from the free packet list and the free buffer list;

testing the contents of the release queue.

14. The line adapter according to claim 5 wherein each list pointer (packet pointer (P_PTR) or queue pointer (Q_PTR)) comprises:

a first field (LID) for identifying the list;

a second field for identifying a next pointer (TAIL) to stack in said list;

a third field for identifying a first pointer (HEAD) stacked in said list.

15. The line adapter according to claim 14 wherein said means for queuing comprises:

means for incrementing the TAIL field of the list pointer;

means for simultaneously storing the pointer identified by the TAIL field in the list identified by the LID field;

means for generating a list full indicator.

16. The line adapter according to claim 14 wherein said means for dequeuing comprises:

means for incrementing the HEAD field of the list pointer;

means for simultaneously reading the pointer identified by the HEAD field, in the list identified by the LID field;

means for generating a list empty indicator.

17. The line adapter according to claim 15 wherein said means for queuing further includes a means for testing said list full indicator, and said means for dequeuing further includes a means for testing said list empty indicator.

18. The line adapter according to claim 1 wherein said programmable processing means includes:

an arithmetical and logical unit,

a register file,

a sequencer,

an instruction file,

a direct memory access controller module, and

a physical memory address generator.

19. A routing method in a line adapter for a packet switching node in a communication network, including programmable processing means (SPP) for receiving and transmitting data packets of fixed or variable length to one or more outputs, said routing method comprising the steps of:

buffering each data packet in one or more buffers in a first storing means;

identifying said buffers;

queuing, in a second storing means, said buffer identifiers in buffer lists, each buffer list identifying a data packet;

identifying said buffer lists;

queuing buffer list identifiers in packet lists in said second storing means, each packet list identifying a queue;

identifying said packet lists;

processing a routing header of each data packet;

associating with each output an output queue for stacking the packet list identifiers of the data packets to transmit on said outputs;

routing the data packets to one or a plurality of outputs;

said processing the routing header step further comprising the step of determining and selecting the output queue corresponding to the destination of each data packet;

said routing step further comprising the steps of:

copying the buffer list identifier of the data packet to transmit in the output queue corresponding to said selected output;

handling each data request for said outputs in real time;

releasing said buffers in said first storing means, and said buffer identifiers and buffer list identifiers from said second storing means;

said handling each data request step including the step of handling independently the output queues by

dequeuing in parallel the packet identifiers requested by the outputs; and

dequeuing the buffer identifiers stacked in the buffer list identified by said packet identifiers.

* * * * *

US005684806A

United States Patent [19]

Akiyama

[11] Patent Number: 5,684,806

[45] Date of Patent: Nov. 4, 1997

[54] **COMMUNICATION APPARATUS FOR TDMA SYSTEM**

[75] Inventor: **Keiji Akiyama**, Tokyo, Japan

[73] Assignee: **Sony Corporation**, Tokyo, Japan

[21] Appl. No.: **492,499**

[22] Filed: **Jun. 20, 1995**

[30] **Foreign Application Priority Data**

Jun. 29, 1994 [JP] Japan 6-148105

[51] **Int. Cl.**[6] **H04J 3/12**

[52] **U.S. Cl.** **370/522**; 370/523; 370/528

[58] **Field of Search** 370/111, 110.1, 370/110.4, 118, 13, 17, 112, 68.1, 95.1, 95.3, 465, 498, 522, 523, 524, 525, 526, 527, 528, 529

[56] **References Cited**

U.S. PATENT DOCUMENTS

4,245,340	1/1981	Landry	370/111
4,330,858	5/1982	Choquet	370/111
4,729,022	3/1988	Shibuya et al.	370/111
4,730,312	3/1988	Johnson et al.	370/110.1
5,440,542	8/1995	Procter et al.	370/111
5,507,006	4/1996	Knight	370/111

FOREIGN PATENT DOCUMENTS

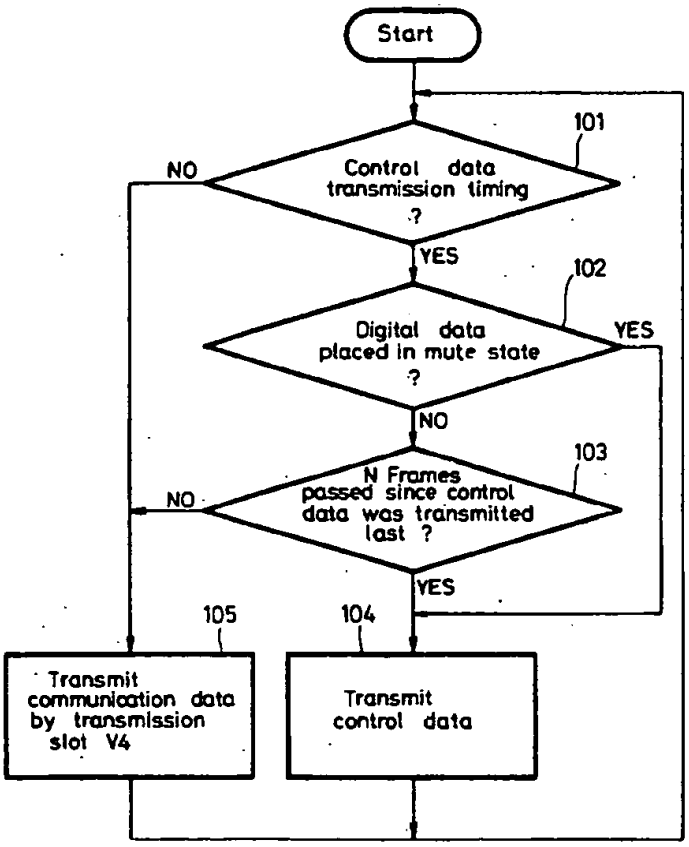
3130176	2/1983	Germany .
2217955	1/1989	United Kingdom .

Primary Examiner—Wellington Chin
Assistant Examiner—Huy D. Vu
Attorney, Agent, or Firm—Jay H. Maioli

[57] **ABSTRACT**

A communication system includes a master station connected to a telephone line network and a plurality of remote stations for transmitting and receiving digital data to and from the master station through a radio transmission line. Among the master station and the remote stations, one channel is divided into a plurality of slots. One of the plurality of slots is assigned as a control slot to transmit and receive data in a time-division manner. The master station transmits control data to the remote stations by the control slot. The master station transmits control data by the control slot at every predetermined period and also transmits data concerning a communication by the control slot during a predetermined period.

11 Claims, 6 Drawing Sheets



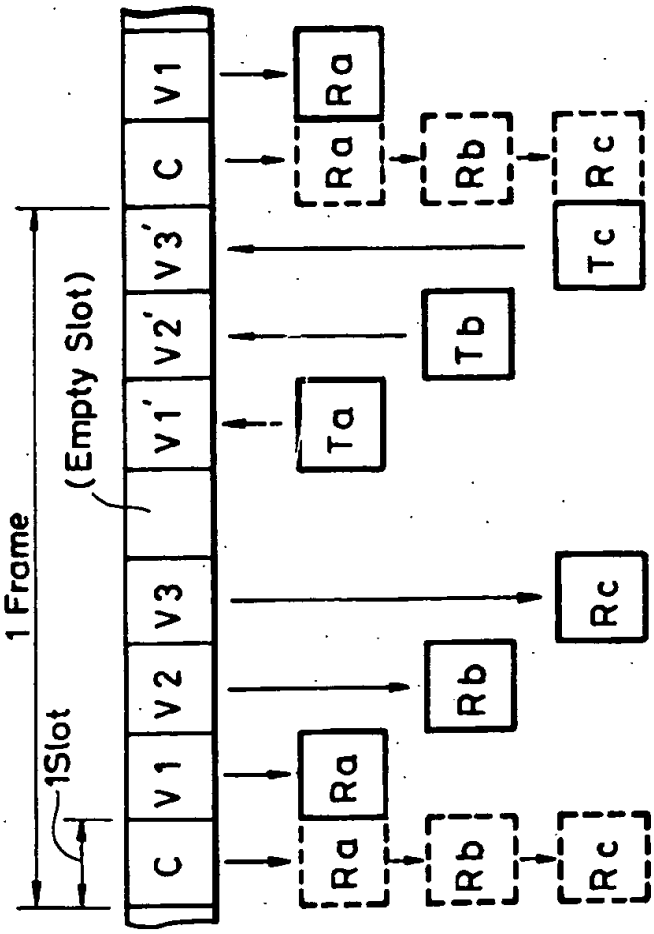


FIGURE 1A Slot Arrangement of Master Station

FIGURE 1B Transmission and Reception Timings of Remote Station a

FIGURE 1C Transmission and Reception Timings of Remote Station b

FIGURE 1D Transmission and Reception Timings of Remote Station c

(PRIOR ART)

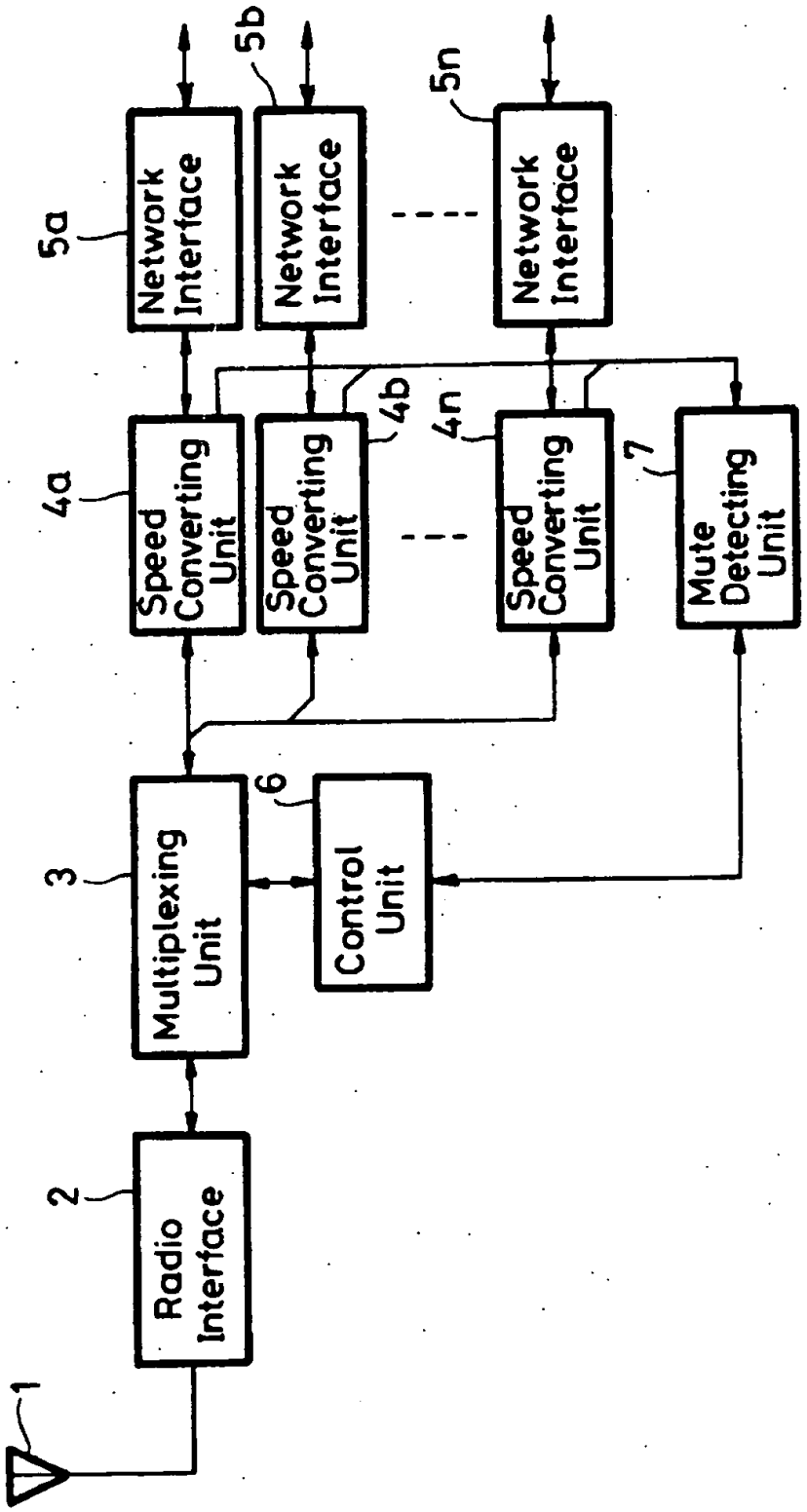

5a
5b
5n
Network Interface
Network Interface
Network Interface
4a
4b
4n
Speed Converting Unit
Speed Converting Unit
Speed Converting Unit
7
Mute Detecting Unit
3
Multiplexing Unit
6
Control Unit
2
Radio Interface
1

FIGURE 2

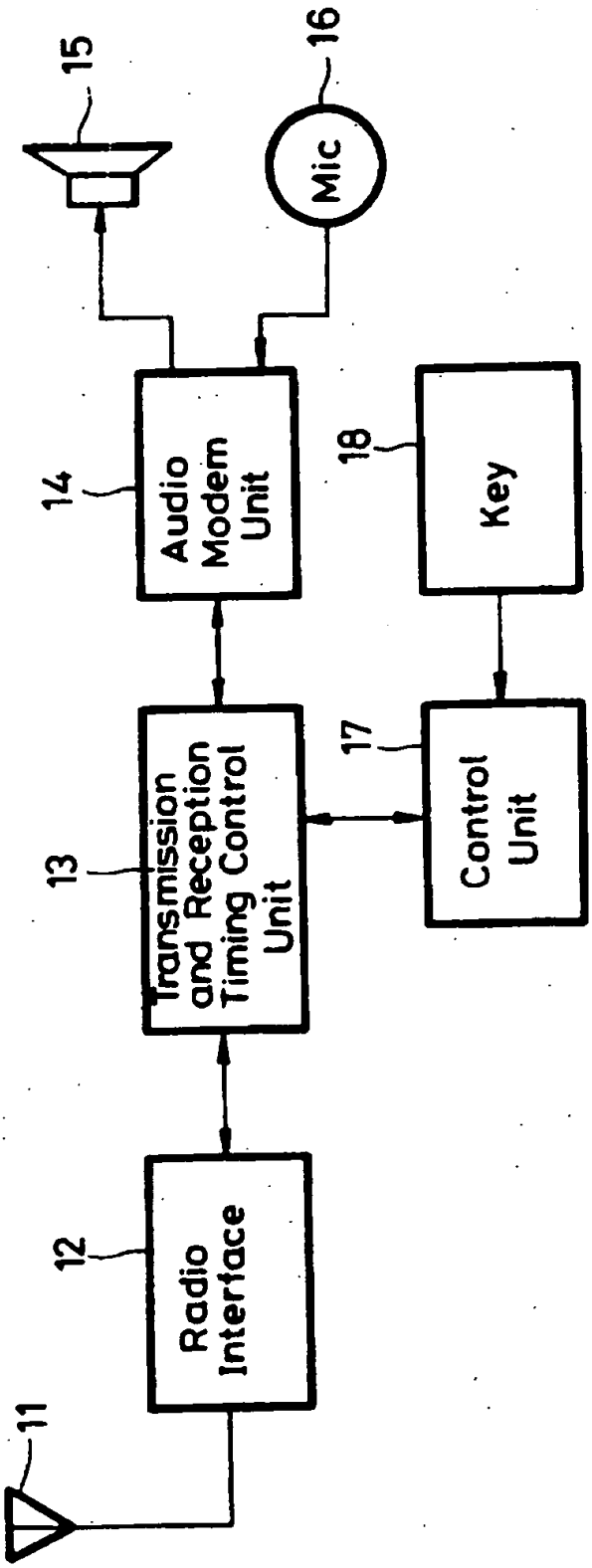
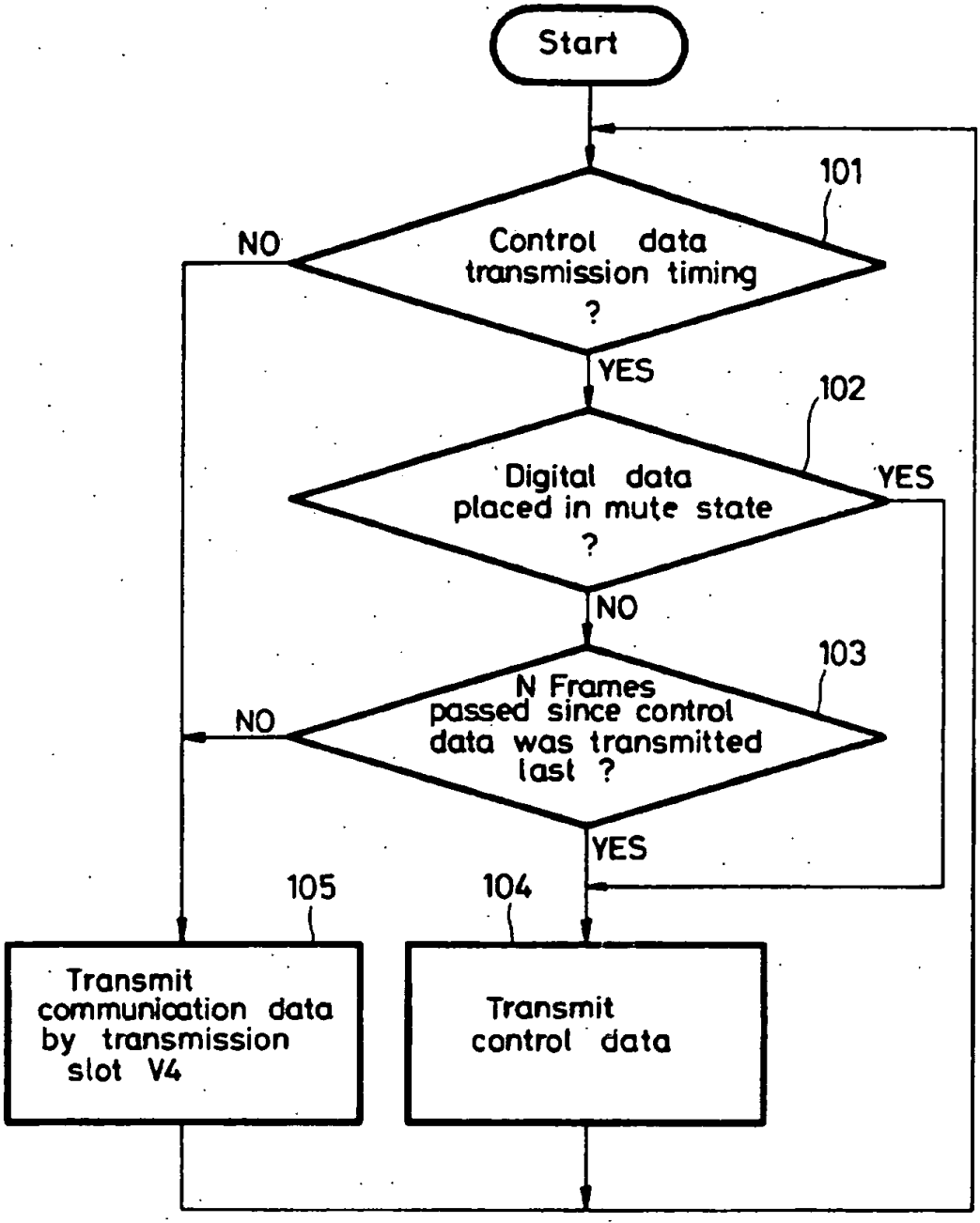

15
16
Mic
14
Audio Modem Unit
18
Key
13
Transmission and Reception Timing Control Unit
17
Control Unit
12
Radio Interface
11

FIGURE 3

FIGURE 4


Start
101
Control data transmission timing ?
NO
YES
102
Digital data placed in mute state ?
YES
NO
103
N Frames passed since control data was transmitted last ?
NO
YES
105
104
Transmit communication data by transmission slot V4
Transmit control data

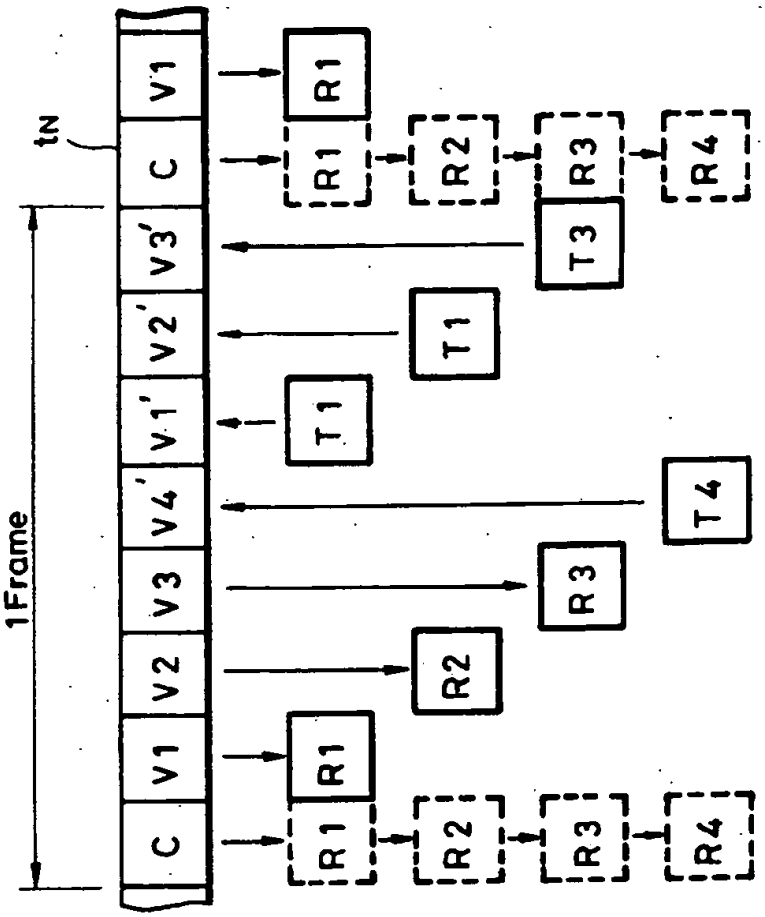


FIGURE 5A Slot Arrangement of Master Station

FIGURE 5B Transmission and Reception Timings of Remote Station M1

FIGURE 5C Transmission and Reception Timings of Remote Station M2

FIGURE 5D Transmission and Reception Timings of Remote Station M3

FIGURE 5E Transmission and Reception Timings of Remote Station M4

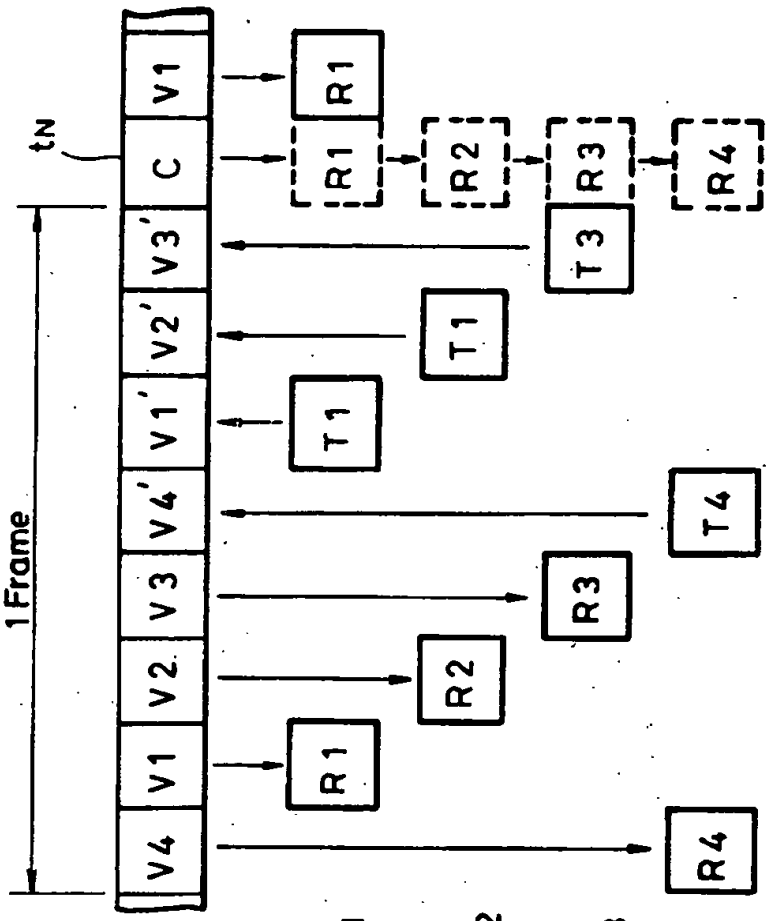


FIGURE 6A Slot Arrangement of Master Station

FIGURE 6B Transmission and Reception Timings of Remote Station

FIGURE 6C Transmission and Reception Timings of Remote Station

FIGURE 6D Transmission and Reception Timings of Remote Station

FIGURE 6E Transmission and Reception Timing of Remote Station

COMMUNICATION APPARATUS FOR TDMA SYSTEM

BACKGROUND

1. Field of the Invention

The present invention relates to a transmission and reception apparatus and, more particularly, is directed to a transmission and reception apparatus for transmitting and receiving data based on a TDMA system.

2. Background of the Invention

A communication system called a TDMA (time division multiple access) system has heretofore been put into practical use. According to the TDMA system, a communication can be made by a slot arrangement such as shown in FIGS. 1A to 1D. In a radio telephone system based on the TDMA system, for example, a master station serving as a base station and a remote station serving as a terminal device can communicate with each other through a radio transmission line. When the remote station and the master station communicate with each other via the radio transmission line, a telephone communication can be made between the person being called who is connected through the telephone line network to the master station and the remote station. In the case of a communication system based on the TDMA system, a single master station and a plurality of remote stations can communicate with one another by using one channel (single frequency). As shown in FIG. 1A, one frame, which is a unit for making a communication, is divided into a plurality of slots (8 slots) on the master station side. In this case, one frame has a duration of 5 milliseconds and one slot has a duration of 0.625 millisecond. Communication is carried out by repeating the units.

When one frame is composed of 8 slots, up to three remote stations a, b, c can be connected simultaneously to the master station by using one channel. Specifically, the first slot of one frame is assigned to a control slot C during which the master station transmits control data to the remote stations. The control data is used to carry out a control, such as assigning slots for transmitting communication data and individually accessing the remote stations. Accordingly, the control data that is transmitted from the master station during the control slot C is received by the remote station which communicates with the master station, i.e., any one of the remote stations a, b, c shown in FIGS. 1B, 1C and 1D.

As shown in FIGS. 1B, 1C and 1D, the three remote stations a, b, c are simultaneously connected to the master station. Three slots V1, V2, V3 which follow the control slot C are slot periods during which communication data is transmitted from the master station to the three remote stations a, b, c. The communication data are results from converting an audio signal of 5 milliseconds into digital data is compressed so that the digital data of 5 milliseconds can be transmitted during one slot period. The three remote stations a, b, c carry out receptions Ra, Rb, Rc during corresponding slot periods as shown in FIGS. 1B, 1C and 1D.

Then, the next one slot after slot V3 is not used i.e. it is, an empty slot. The master station receives communication data from the three remote stations a, b, c during the last three slots V1', V2', V3'. Accordingly, the three remote stations a, b, c carry out transmissions Ta, Tb, Tc of communication data during corresponding slot periods as shown in FIGS. 1B, 1C and 1D. Also in this case, communication data are results from converting an audio signal of 5 milliseconds into digital data that is compressed so that the digital data of 5 milliseconds can be transmitted during one slot period.

According to the circumstances, control data is received during the empty slot provided between the transmission slot V3 and the reception slot V1' of the master station. The transmission of control data in the slot assigned to the control slot C is not carried out for every frame. But, it is customary that such transmission is carried out only once during a period ranging from several frames to several tens of frames.

If one channel is divided into slots as described above, then communication data is transmitted and received among one master station and the three remote stations by using one channel (i.e., single transmission frequency). Therefore, it is possible for the master station to simultaneously communicate with many terminal devices with fewer channels.

Even though the master station can be connected to a plurality of remote stations by using one channel in a multiple access fashion, the number of remote stations that can be connected to the master station in a multiple access fashion is unavoidably limited. In the example shown in FIGS. 1A through 1D, only three remote stations can be connected to the master station through one channel at maximum. On the other hand, there is a demand for improving the efficiency of using a transmission channel by increasing the number of remote stations connected to the master station through one channel.

In order to increase the number of remote stations that can be connected to the master station via one channel, it has been proposed that the number of slots provided in one frame be increased. In this case, however, if the number of slots is increased, then either the duration of the frame period is extended or the duration of the slot period is reduced. For the latter option, however, it is frequently observed that audio data cannot be compressed satisfactorily.

SUMMARY OF THE INVENTION

It is therefore an object of the present invention to provide a transmission and reception apparatus in which the above-mentioned problem can be solved.

It is another object of the present invention to provide a communication system in which the above-mentioned problem can be solved.

According to the present invention, there is provided a transmission and reception apparatus. The transmission and reception apparatus divides one channel into a plurality of slots. One of the plurality of slots is assigned as a control slot in which control data is processed in a time division multiplex fashion and transmitted and received together with data concerning a communication. The transmission and reception apparatus includes a transmission and reception unit and a control unit. The transmission and reception unit processes data in a time-division multiplex fashion and transmits the data thus processed. Also, the transmission and reception unit receives transmitted data. The control unit controls an operation of the transmission and reception unit. The control unit transmits data concerning a communication by the control slot at a time to transmit control data during a predetermined period provided after the control data has been transmitted by the control slot.

According to the present invention, there is provided a communication system which includes a master station connected to a telephone line network and a plurality of remote stations which transmit and receive digital data to and from the master station via a radio transmission line. Among the master station and the remote stations, one channel is divided into a plurality of slots and one of the

plurality of slots is assigned as a control slot during which digital data is transmitted and received in a time division manner. Also, control data is transmitted from the master station to the remote stations by the control slot. The master-station transmits control data by the control slot at every predetermined period also transmits data concerning a communication by the control slot during a predetermined period.

According to the present invention, data concerning a communication is transmitted by using the control slot when control data is not transmitted by the control slot. Thus, it is possible to increase the number of remote stations that can be connected to the master station by using one channel.

BRIEF DESCRIPTION OF THE DRAWINGS

FIGS. 1A through 1D are timing charts showing an example of an arrangement of slots based on the TDMA system;

FIG. 2 is a block diagram showing an arrangement of a master station according to an embodiment of the present invention;

FIG. 3 is a block diagram showing an arrangement of a remote station according to the embodiment of the present invention;

FIG. 4 is a flowchart to which references will be made in explaining the condition of when an inventive slot is in use;

FIGS. 5A through 5E are timing charts showing an arrangement of slots provided when it is detected that communication data transmitted by a slot V4 is placed in the mute state according to the embodiment of the present invention; and

FIGS. 6A through 6E are timing charts showing an arrangement of slots provided when it is detected that communication data transmitted by the slot V4 is not placed in the mute state according to the embodiment of the present invention.

DESCRIPTION OF THE INVENTION

A transmission and reception apparatus according to an embodiment of the present invention will hereinafter be described in detail with reference to the drawings.

In this embodiment, the transmission and reception apparatus according to the present invention is applied to a radiotelephone system in which a signal, which was converted to digital data, is transmitted and received among a master station serving as a base station and terminal stations serving as remote stations in a TDMA system fashion. FIG. 2 shows in block form an arrangement of an inventive master station.

As shown in FIG. 2, an antenna 1 is connected to a radio interface 2. The radio interface 2 demodulates a received signal received at the antenna 1 or modulates the received signal so that the signal can be transmitted or received satisfactorily. This radio interface 2 is connected to a multiplexing unit 3. The multiplexing unit 3 separates data inserted into respective slots of the received signal and synthesizes data so that data is inserted into the respective slots of a transmitted signal. An arrangement of the slots will be described later on.

A plurality of speed converting units 4*a*, 4*b*, . . . , 4*n* are connected to the multiplexing unit 3. In this embodiment, the number of the speed converting units 4*a* through 4*n* that can be connected to the multiplexing unit 3 is selected to be 4 so that there are prepared 4 line networks. Each of the speed converting units 4*a* through 4*n* carries out a speed conversion processing to convert intermittent received data extracted from respective slots to consecutive data. Moreover, each of the speed converting units 4*a* through 4*n* carries out a speed conversion processing to convert transmission data to intermittent data so that the transmission data can be inserted into the respective slots. Network interfaces 5*a*, 5*b*, . . . , 5*n* are connected to the speed converting units 4*a* through 4*n*, respectively. Each of the network interfaces 5*a*, 5*b*, . . . , 5*n* is connected to a digital telephone line network, such as an ISDN (integrated service digital network) line network. The network interfaces 5*a* through 5*n* convert a clock rate in order to convert data output thereto from the speed converting units 4*a* through 4*n* to data transmitted to the telephone line network. Moreover, the network interfaces 5*a* through 5*n* convert the clock rate in order to convert data received from the telephone line network to radio transmission data by the radio interface 2.

The processing in each circuit provided within the master station shown in FIG. 2 is executed at a synchronized timing under the control of a control unit 6.

Data supplied to the speed converting units 4*a* through 4*n* are also supplied to a mute detecting unit 7. The mute detecting unit 7 determines whether or not communication data transmitted from the telephone line network side to the network interfaces 5*a* to 5*n* at every network are placed in the mute state. As a method used by the mute detecting unit 7 to determine whether or not communication data is placed in the mute state, there are enumerated a method of detecting a mute state code added when a transmitted audio signal is placed in the mute state and a method of detecting based on the state of transmitted audio data whether or not communication data is placed in the mute state. In this case, when the four speed converting units 4*a* through 4*n* are all in use, or four networks are connected to the master station, the mute detecting unit 7 detects the mute state of the network connected last. When the mute detecting unit 7 detects the mute state, a detected result of the mute detecting unit 7 is supplied to the control unit 6.

The control unit 6 supplies the control data to the multiplexing unit 3 at a predetermined time to enable the control data to be transmitted to the remote station side by a predetermined slot. Also, control data of up-link which is transmitted from the remote station by a predetermined slot and which is supplied to the multiplexing unit 3 is supplied to the control unit 6.

The control unit 6 carries out a communication control processing such that up to four remote stations, i.e., four line networks can communicate with one another simultaneously by one channel through radio waves. When the four line networks are simultaneously in use, if it is determined by the mute detecting unit 7 that communication data transmitted from the telephone line network side to the speed converting unit 4*n* of one line network is placed in the mute state, then the control unit 6 supplies control data, which is to be transmitted to the remote station, to the multiplexing unit 3. However, when a transmission of control data is stopped during a predetermined period, even if it is determined by the mute detecting unit 7 that communication data is not placed in the mute state, then the control unit 6 supplies the control data to the multiplexing unit 3 to transmit the same by a predetermined slot. This processing will be described later on.

FIG. 3 shows in block form an arrangement of a remote station according to the embodiment of the present invention. As shown in FIG. 3, an antenna 11 is connected to a radio interface 12. The radio interface 12 demodulates or

modulates a received signal received at the antenna 11 so that the signal can be transmitted or received satisfactorily upon radio communication. The radio interface 12 is connected to a transmission and reception timing control unit 13. The transmission and reception control unit 13 extracts only a signal of a predetermined slot from a received signal and carries out a necessary processing based on the TDMA system in which a transmission signal is transmitted at a timing of a predetermined slot. The transmission and reception timing control unit 13 controls the timing under the control of a control unit 17 which controls the communication of the remote station. In this case, the control unit 17 carries out a control operation based on control data extracted from the received data by the transmission and reception timing control unit 13, i.e., control data transmitted thereto from the master station. When the transmission and reception timing control unit 13 cannot receive control data, the transmission and reception timing control unit 13 memorizes control data received last and continues carrying out a control operation based on the control data thus stored therein.

The transmission and reception timing control unit 13 is connected to an audio MODEM (modulator and demodulator) unit 14. The audio MODEM unit 14 carries out a demodulation processing wherein digital audio data is extracted from the received data and converted into an analog audio signal and a processing wherein the analog audio signal is converted to digital audio data and transmitted as a transmission signal. A speaker 15 and a microphone (MIC) 16 are connected to the audio MODEM unit 14. The audio signal that has been demodulated by the audio MODEM unit 14 is emanated from the speaker 15 and a sound picked up by the microphone 16 is supplied to the audio MODEM unit 14 as an audio signal.

Therefore, in the inventive remote station, when audio data supplied from the transmission and reception timing control unit 13 to the audio MODEM unit 14 is interrupted temporarily, the audio signal supplied to the speaker 15 is temporarily placed in the mute state until the transmission and reception timing control unit 13 starts supplying audio data to the audio MODEM unit 14.

A key 18 such as a dial key is connected to the control unit 17 which carries out communication control in this remote station. The transmission and reception apparatus can make an outgoing call or receive an incoming call as a telephone set when the key 18 is operated.

A communication state among the inventive master station and the inventive remote stations will be described below with reference to a flowchart of FIG. 4 and communication timing charts shown in FIGS. 5A to 5E and FIGS. 6A to 6E.

Initially, a slot arrangement applied to the communication according to this embodiment will be described below. According to this embodiment, as shown in FIGS. 5A to 5E or FIGS. 6A to 6E, one frame is composed of 8 slots. Fundamentally, 4 slots of the first half of one frame are assigned to transmission slots of the master station and 4 slots of the second half of one frame are assigned to the reception slots of the master station. This frame structure is repeated and one frame has a duration of 5 milliseconds.

As shown in FIGS. 5A through 5E, the first one slot of one frame is assigned to the control slot C wherein the master station transmits control data to the remote stations when predetermined conditions are satisfied, e.g., four remote stations are connected to the master station as will be described later on. Depending on the condition of the network that is in use, as shown in FIGS. 6A to 6E, it is frequently observed that this control slot C becomes a transmission slot V4 wherein control data is transmitted from the master station to the remote stations. When this transmission slot V4 is in use, as shown in FIGS. 5A to 5E and FIGS. 6A to 6E, the first slot of the 4 slots of the second half of one frame is employed as a reception slot V4' corresponding to the transmission slot V4.

Conditions by which the slots are set as described above will be described below. Initially, when there are three remote stations that are connected to the master station (three remote stations will hereinafter be referred to as "remote stations M1, M2, M3"), transmission and reception are carried out among the remote stations M1, M2, M3 shown in FIGS. 5B, 5C, 5D and FIGS. 6B, 6C, 6D by using transmission slots V1, V2, V3 and the reception slots V1', V2', V3' shown in FIGS. 5A and 6A. In that case, the master station transmits control data to the remote stations by using the control slots C of the starting portions of the respective frames to control the communication states of the respective remote stations M1, M2, M3 on the basis of the control data. The above-mentioned operations are the same as those shown in FIGS. 1A through 1D.

According to this embodiment, in the communication state with this frame structure, another remote station M4 can be connected to the master station (i.e., four remote stations M1, M2, M3, M4 in total can be connected to the master station). When the four remote stations M1, M2, M3, M4 are connected to the master station, the control data may be transmitted by using the control slot C or the control slot C may be used as the transmission slot V4 through which control data is transmitted from the master station to the remote station M4. This decision will be described below with reference to the flowchart shown in FIG. 4.

As shown in FIG. 4, following the start of operation, it is determined in decision step 101 whether or not a timing becomes a control data transmission timing. In other words, according to the inventive communication system, since it is sufficient that control data is transmitted once per 20 frames, it is determined in decision step 101 whether or not a timing becomes a transmission timing at which the control slot C is assigned once per 20 frames. If a timing is not the transmission timing of this control slot C as represented by a NO at decision step 101, then the processing proceeds to step 105, whereat the control slot C is used as the transmission slot V4 for the remote station M4 to transmit communication data. Moreover, the remote station M4 shown in FIG. 6E receives communication data by a reception slot R4. At that time, the reception slot V4' corresponding to the transmission slot V4 also is set and a transmission of data from the remote station M4 by using the transmission slot T4 is executed at this timing.

If on the other hand the timing becomes the control data transmission timing as represented by a YES at decision step 101, then the processing proceeds to the next decision step 102. It is determined in decision step 102 whether or not digital data transmitted from the telephone line network side of the network connected last is placed in the mute state. At that time, it is determined on the basis of detected data from the mute detecting unit 7 whether or not the digital data is placed in the mute state. If the digital data is placed in the mute state as represented by a YES at decision step 102, then the processing proceeds to step 104, whereat control data is transmitted as the control slot C. The frame structure provided at that time becomes such one as shown in FIGS. 5A through 5E. The control data used at that time is data that is transmitted to all of the remote stations M1 to M4.

If on the other hand the transmitted digital data is not placed in the mute state as represented by a NO at decision step **102**, then the processing proceeds to the next decision step **103**. In decision step **103**, it is determined whether or not N frames (e.g., 40 frames) are passed since the control data was transmitted last. The N frames have a period corresponding to a limit time wherein the remote stations M1 to M4 can maintain a communication among them and the master station on the basis of the previously-received control data, i.e., the remote stations M1 to M4 can maintain a frame synchronization. Therefore, if control data were not transmitted once at least in the N frames, the remote stations could not communicate with the master station.

If the N frames are passed as represented by a YES at decision step **103**, then the processing proceeds to step **104**, whereat the control data is transmitted as the control slot C. If on the other hand the N frames are not passed as represented by a NO at decision step **103**, then the processing proceeds to step **105**, whereat the control slot C is set to the transmission slot V4 of the remote station M4 to transmit communication data.

In any case, a slot provided between the transmission slot V3 and the reception slot V1' of the master station is set to the reception slot V4' from the remote station M4 shown in FIGS. **5E** and **6E** to receive communication data transmitted by the transmission slot T4 of the remote station M4.

The state shown at the timing t_N shown in FIGS. **5A** through **5E** shows the state that the control data is transmitted by the control slot C even when the digital data is not placed in the mute state after it was determined in decision step **103** that the timing becomes the timing having the period corresponding to the limit time in which a communication can be maintained at this timing t_N.

As described above, according to the inventive frame structure, the master station can communicate with one remote station by using the slot assigned to transmit control data. Thus, it is possible to increase the number of the remote stations that can be connected to the master station by one without varying the frame period and the like. In this case, if the control data is transmitted when the communication data transmitted to the remote station which uses this slot is placed in the mute state, then the control data is transmitted so long as no trouble occurs even though the transmission of data of one slot is failed during communication data is transmitted between the master station and the remote station. Therefore, it is possible to efficiently transmit both the communication data concerning audio data and the control data by using the control slot. Incidentally, since the mute state of communication data appears frequently upon ordinary conversation, there is then the large possibility that control data can be transmitted at the predetermined period.

If the communication data is not placed in the mute state, then when the timing becomes the limit of time whereat the communication can be maintained by the control data transmitted last, control data is forced to be transmitted. Therefore, the communication states among the remote stations can be maintained satisfactorily. In this case, although communication data concerning audio data is interrupted once per N frames during one frame period in the remote station M4 which was connected to the master station last, in the remote station according to this embodiment, an audio signal reproduced at that time is placed in the mute state only during one frame period, i.e., 5 milliseconds. This one frame period is not long enough for the user to identify the mute state in actual practice. Therefore, this one frame period does not hinder the user from hearing communication data substantially.

When control data or communication data is transmitted by one slot, since corresponding identification codes are added to the control data and the communication data, even if the two data of the control data and the communication data are provided in the mixed state like this embodiment, it is possible to easily discriminate the communication data and the control data one from another by the remote station.

While the four remote stations M1 to M4 are connected to the master station by using one transmission channel as described above, when any one of the slots becomes disabled in use after a communication done by the remote station which uses any of the slots V1 and V1', V2 and V2', V3 and V3' was finished, it is preferable that a communication which uses the slots V4, V4' is moved to the disabled slot. If the slot in use is moved as described above, then it is possible to prevent communication data, i.e., a sound output from the speaker **15** from being interrupted temporarily.

The used slot can be moved by transmitting corresponding control data from the master station to the remote station on the basis of the control done by the control unit **6** of the master station.

When control data such as a connection request signal is transmitted from the respective remote stations to the master station by using the same channel, it is sufficient to use the first slot V4' of the second half slots of one frame, for example. In this case, since the connection request signal need not be transmitted when a connection between the remote station and the master station is completed, the master station need not receive control data from the remote station under the condition that the four remote stations are all connected to the master station. Therefore, it is sufficient that the master station receives only communication data from the remote station under the condition that the control slot is used as the slot V4'. Thus, the four remote stations and the master station can communicate with one another satisfactorily.

Further, while one frame is composed of 8 slots to multiplex four telephone networks as described above, the present invention is not limited thereto. It is needless to say that the control slot may be used as a communication slot when one frame is composed of several slots other than 8 slots.

Furthermore, while a transmission signal modulation system between the master station and the remote station and the like have not been described so far in detail, it is needless to say that the present invention can be applied to a variety of communication systems which transmit and receive data of slot arrangement.

Having described a preferred embodiment of the invention with reference to the accompanying drawings, it is to be understood that the invention is not limited to that precise embodiment and that various changes and modifications could be effected therein by one skilled in the art without departing from the spirit or scope of the invention as defined in the appended claims.

What is claimed is:

1. A transmission and reception apparatus including a channel divided into a plurality of frames, each of said plurality of frames being subdivided into a plurality of slots, one slot of each of said plurality of frames being assigned as a dual-purpose slot for carrying control data or communication data and the remaining slots of each of said plurality of frames being assigned as communication slots for carrying communication data, wherein control data is transmitted and received together with communication data, said apparatus comprising:

transmitting and receiving means for transmitting data in the form of time-division-multiplexed data and for receiving data transmitted thereto;

control means for controlling an operation of said transmitting and receiving means, said control means controlling said transmitting and receiving means to transmit said communication data in said dual-purpose slot of all subsequent ones of said plurality of frames at a time for transmitting said control data during a predetermined period after said control data is transmitted in said dual-purpose slot of a previously transmitted one of said plurality of frames, said control means controlling said transmitting and receiving means to transmit said control data in said dual-purpose slot when not transmitting said communication data, and said control means controlling said transmitting and receiving means to transmit said communication data in said communication slots in all of said plurality of frames,

whereby each of said plurality of frames includes a dual-purpose slot carrying one of control data and communication data; and

detecting means for detecting a mute state from an output signal supplied thereto by said transmitting and receiving means,

wherein said control means controls said transmitting and receiving means to transmit said control in said dual-purpose slot of all subsequent ones of said plurality of frames while said detecting means detects said mute state.

2. A transmission and reception apparatus according to claim 1, wherein said control means transmits said control data in a dual-purpose slot of one of said plurality of frames within a predetermined period after said control data is last transmitted in a previously transmitted one of said plurality of frames.

3. A transmission and reception apparatus according to claim 1, wherein said control means transmits said control data in a dual-purpose slot of one of said plurality of frames after a predetermined time has elapsed from when said control data is last transmitted in a previously transmitted one of said plurality of frames.

4. A transmission and reception apparatus according to claim 3, wherein said control means mutes an output audio signal when said control means detects that audio data supplied thereto from said transmitting and receiving means has been interrupted.

5. A transmission and reception apparatus including a channel divided into a plurality of frames, each of said frames being subdivided into a plurality of slots, one slot of each of said plurality of frames being assigned as a dual-purpose slot for carrying control data or communication data and the remaining slots of each of said plurality of frames being assigned as communication slots for carrying communication data, wherein control data is transmitted and received together with communication data, said apparatus comprising:

transmitting and receiving means for transmitting data in the form of time-division-multiplexed data and for receiving data transmitted thereto;

control means for controlling an operation of said transmitting and receiving means, said control means controlling said transmitting and receiving means to transmit said control data in said dual-purpose slot of first predetermined ones of said plurality of frames during a first predetermined period and said control means controlling said transmitting and receiving means to transmit said communication data in said dual-purpose slot of second predetermined ones of said plurality of frames during a second predetermined period, whereby the first predetermined ones of the plurality of frames and the second predetermined ones of the plurality of frames include all of the plurality of frames; and

detecting means for detecting a mute state from an output signal supplied thereto by said transmitting and receiving means,

wherein said control means controls said transmitting and receiving means to transmit said control data in said dual-purpose slot of all subsequent ones of said plurality frames while said detecting means detects said mute state.

6. A transmission and reception apparatus according to claim 5, wherein said control means transmits said control data in a dual-purpose slot of one of said plurality of frames within a predetermined period after said control data is last transmitted in a previously transmitted one of said plurality of frames.

7. A transmission and reception apparatus according to claim 6, wherein said control means mutes an output audio signal when said control means detects that audio data supplied thereto from said transmitting and receiving means has been interrupted.

8. A communication system comprising:

a master station connected to a telephone line network; and

a plurality of remote stations for transmitting and receiving digital data to and from said master station through a radio transmission line, wherein

a communication channel between said master station and said remote stations is divided into a plurality of frames, each of said plurality of frames being subdivided into a plurality of slots,

one slot of each of said plurality of frames is assigned as a dual-purpose slot carrying control data or communication data and the remaining slots of each of said plurality of frames being assigned as communication slots for carrying communication data,

said master station transmits control data to said remote stations in said dual-purpose slot of each of first predetermined ones of said plurality of frames during a first predetermined period,

said master station transmits communication data in said dual-purpose slot of each of second predetermined ones of said plurality of frames during a second predetermined period,

whereby the first predetermined ones of the plurality of frames and the second predetermined ones of the plurality of frames include all of the plurality of frames; and

detecting means for detecting a mute state from an output signal of one of said plurality of remote stations,

wherein said master station transmits said control data in said dual-purpose slot of all subsequent ones of said

plurality of frames while said detecting means detects said mute state.

9. A communication system according to claim 8, wherein a remote station communicates with said master station based on a most recently received control data when said remote station does not receive control data subsequent to said most recently received control data from said master station.

10. A communication system according to claim 9, wherein said master station transmits said control data in a dual-purpose slot of one of said plurality of frames after a predetermined period has elapsed from when said control data was last transmitted in a previously transmitted one of said plurality of frames.

11. A communication system according to claim 10, wherein said remote station mutes an output of said master station during a predetermined period when said remote station detects that audio data supplied from said master station has been interrupted.

* * * * *

US005768282A

United States Patent [19]

Ohara et al.

[11] Patent Number: 5,768,282

[45] Date of Patent: Jun. 16, 1998

[54] **NODE PROVIDED WITH FACILITY FOR CHECKING ESTABLISHMENT OF SYNCHRONIZATION**

[75] Inventors: Yasuko Ohara; Hiroshi Yoshida, both of Kawasaki, Japan

[73] Assignee: Fujitsu Limited, Kanagawa, Japan

[21] Appl. No.: 614,946

[22] Filed: Mar. 11, 1996

Related U.S. Application Data

[63] Continuation of Ser. No. 213,254, Mar. 15, 1994, abandoned.

[30] **Foreign Application Priority Data**

Mar. 15, 1993 [JP] Japan 05-053659

[51] **Int. Cl.**[6] **H04J 3/06**

[52] **U.S. Cl.** **370/506**; 370/509

[58] **Field of Search** 370/100.1, 105.1, 370/105, 105.3, 105.4, 108, 84, 112, 110.1, 13, 503, 506, 516, 522, 528, 507, 509, 505, 242, 541, 244, 246, 535, 539, 102; 375/362, 365, 366, 372, 373, 354

[56] **References Cited**

U.S. PATENT DOCUMENTS

4,748,623	5/1988	Fujimoto	370/105
4,856,063	8/1989	McCalmont	380/48
5,113,395	5/1992	Murakami et al.	370/102
5,128,939	7/1992	Takatori et al.	370/100.1
5,155,728	10/1992	Takeuchi et al.	370/100.1
5,172,376	12/1992	Chopping et al.	370/100.1
5,241,543	8/1993	Amada et al.	370/100.1
5,265,096	11/1993	Parruck	370/110.1
5,325,354	6/1994	Hadano	370/112
5,331,630	7/1994	Fujita et al.	370/84
5,335,223	8/1994	Iino	370/108
5,337,334	8/1994	Molloy	375/118
5,343,476	8/1994	Urbansky	370/108

FOREIGN PATENT DOCUMENTS

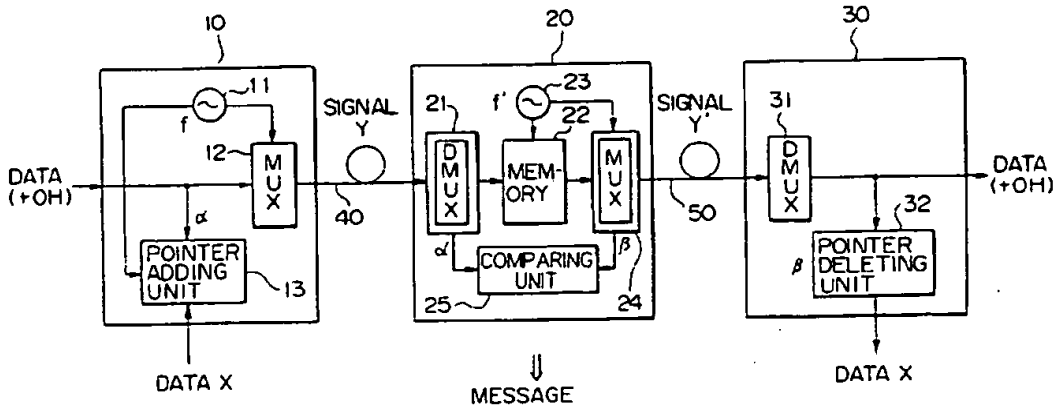
40 12 762 10/1991 Germany .

Primary Examiner—Chau Nguyen
Attorney, Agent, or Firm—Helfgott & Karas, P.C.

[57] **ABSTRACT**

A node in a synchronous communication network performs communication by sending and receiving frames formed by successively adding pointers showing header positions to a plurality of data, wherein a comparison is made between the values of pointers added to data received from an opposing node side and the values of pointers to be added to the data sent from a home node to detect if the two values coincide. Synchronization has been established between the opposing node and the home node when they do.

6 Claims, 13 Drawing Sheets



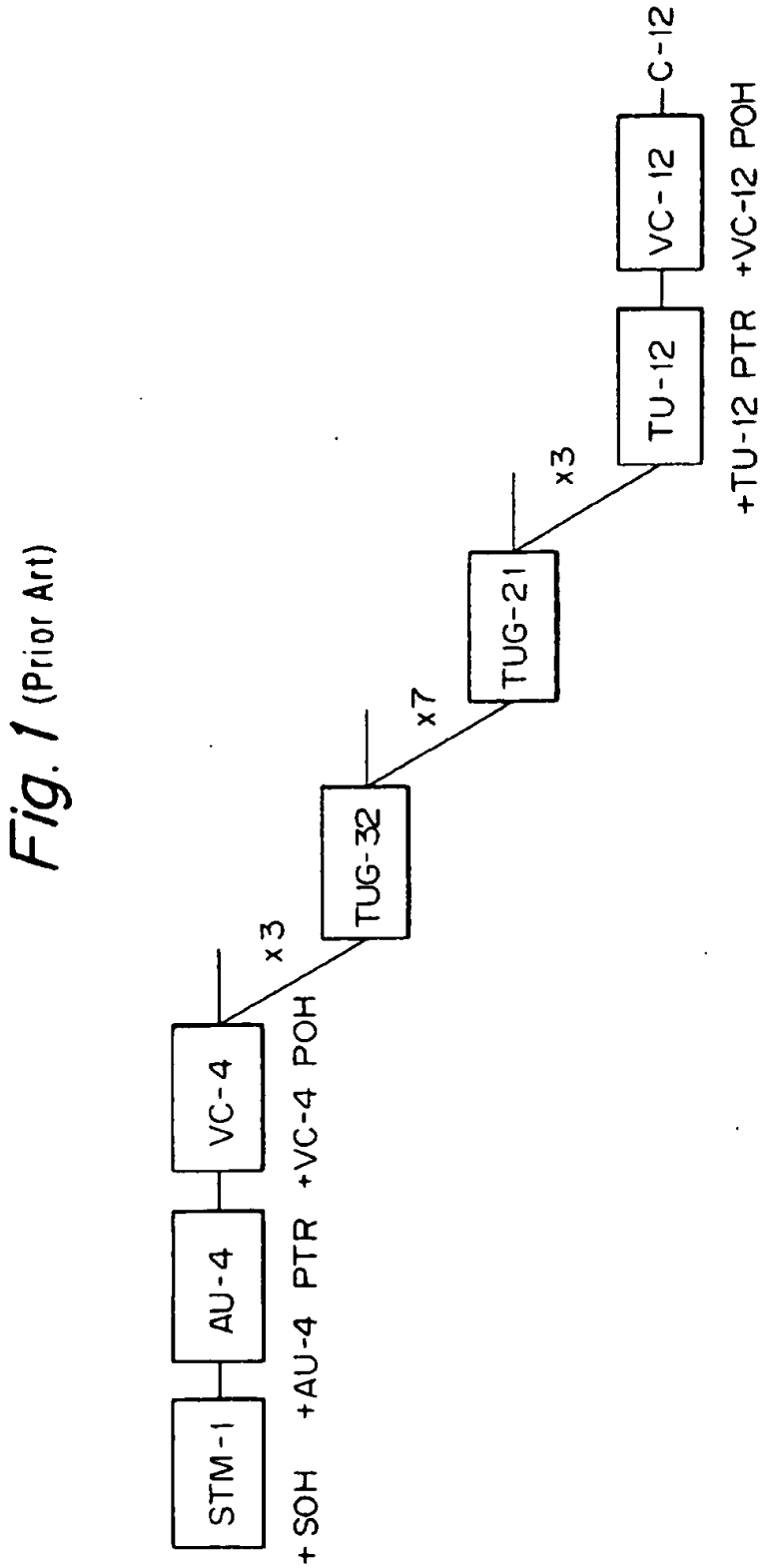


Fig. 1 (Prior Art)

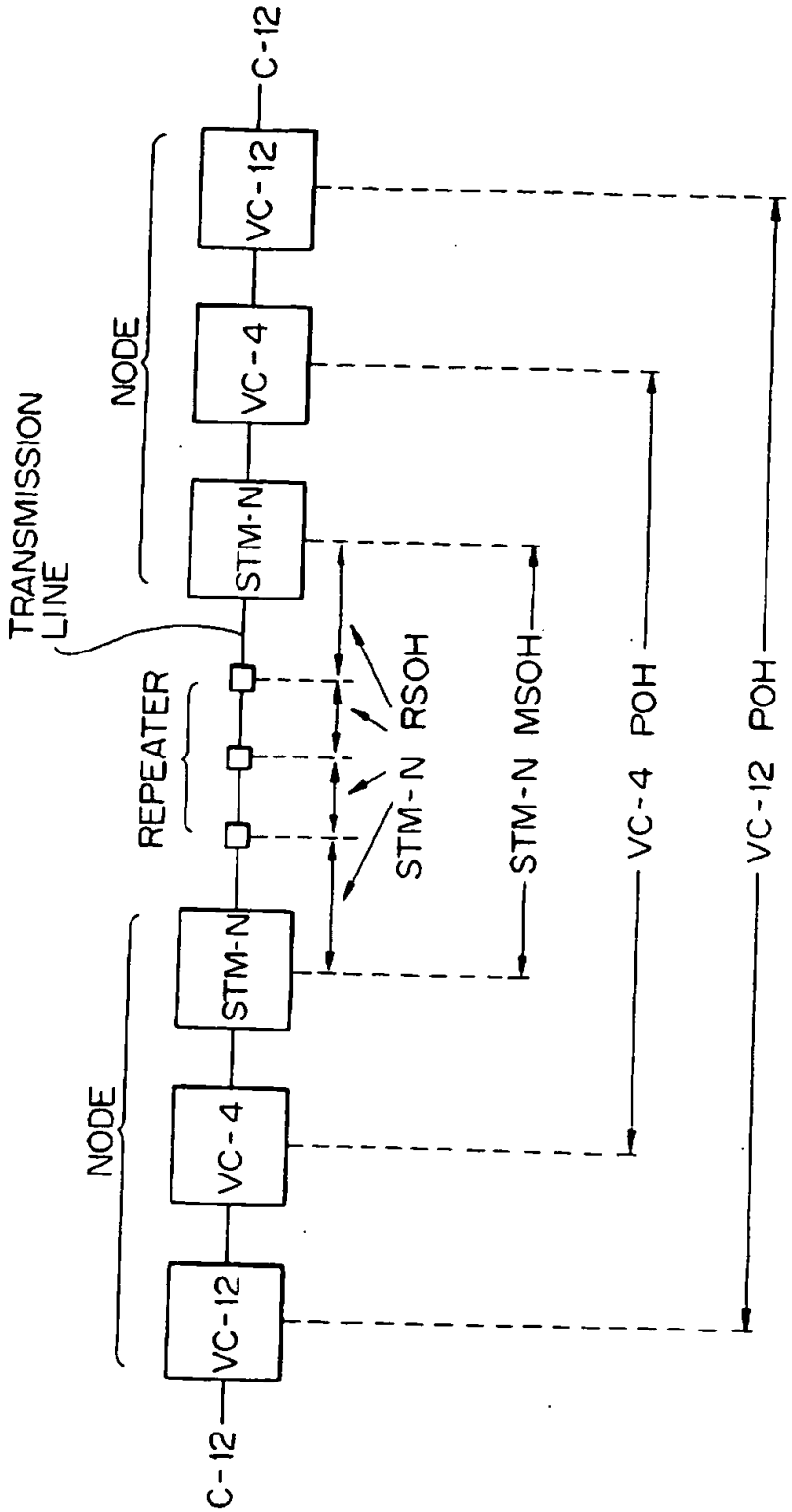


Fig. 2 (Prior Art)

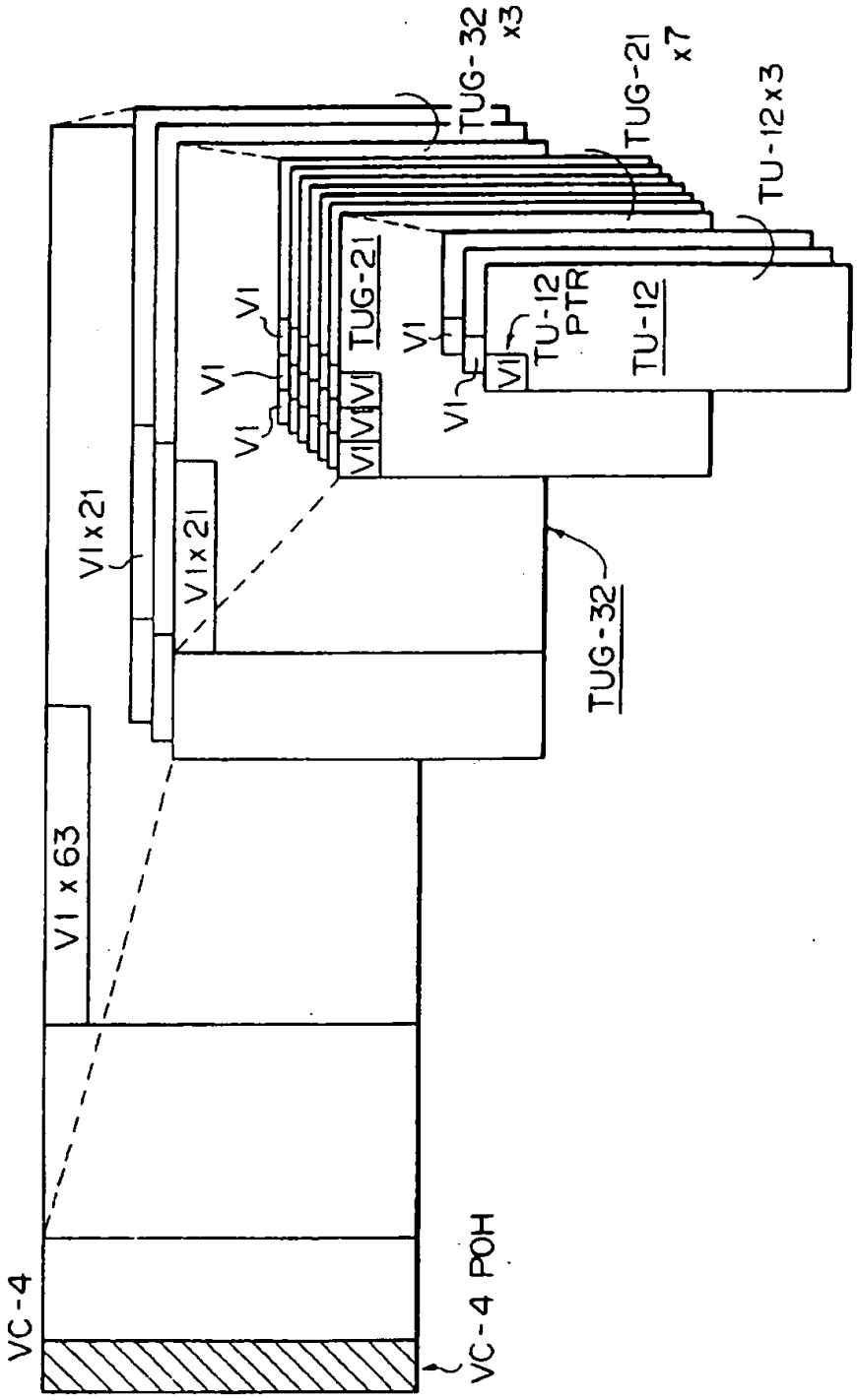


Fig.3 (Prior Art)

Fig. 4 (Prior Art)

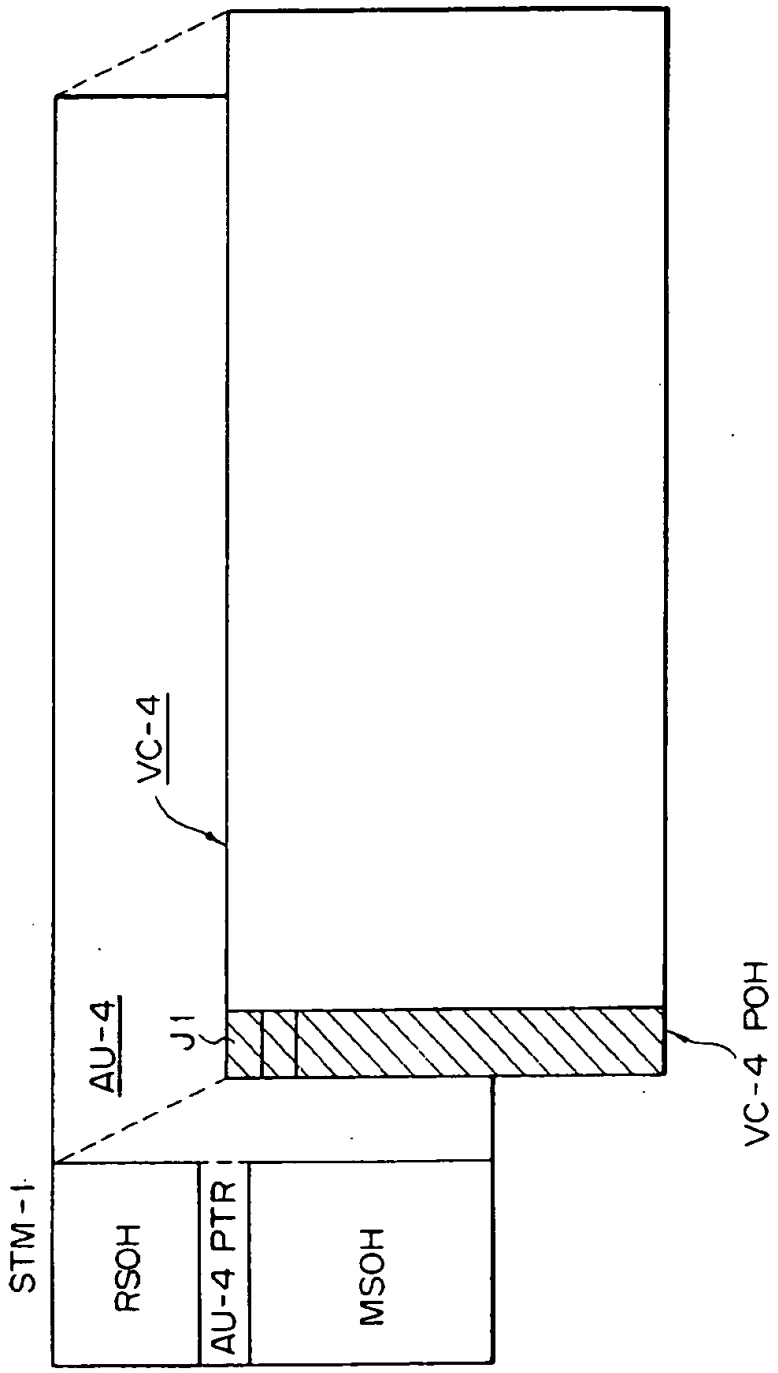


Fig.5A

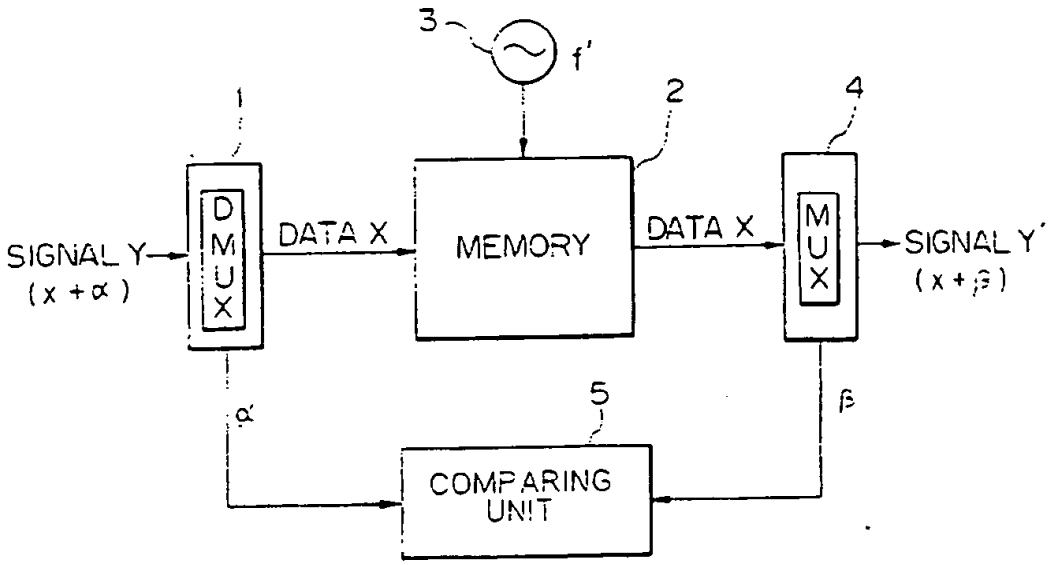

3
f'
1
2
4
SIGNAL Y
(x + α)
DMX
DATA X
MEMORY
DATA X
MUX
SIGNAL Y'
(x + β)
α'
β
5
COMPARING UNIT

Fig.5B

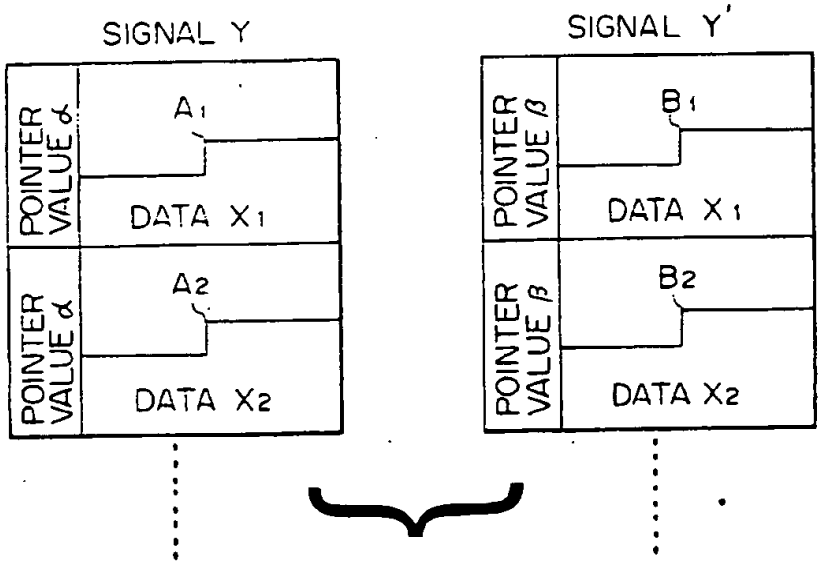

SIGNAL Y
SIGNAL Y'
POINTER VALUE α
A1
DATA X1
POINTER VALUE α
A2
DATA X2
POINTER VALUE β
B1
DATA X1
POINTER VALUE β
B2
DATA X2

Fig.6

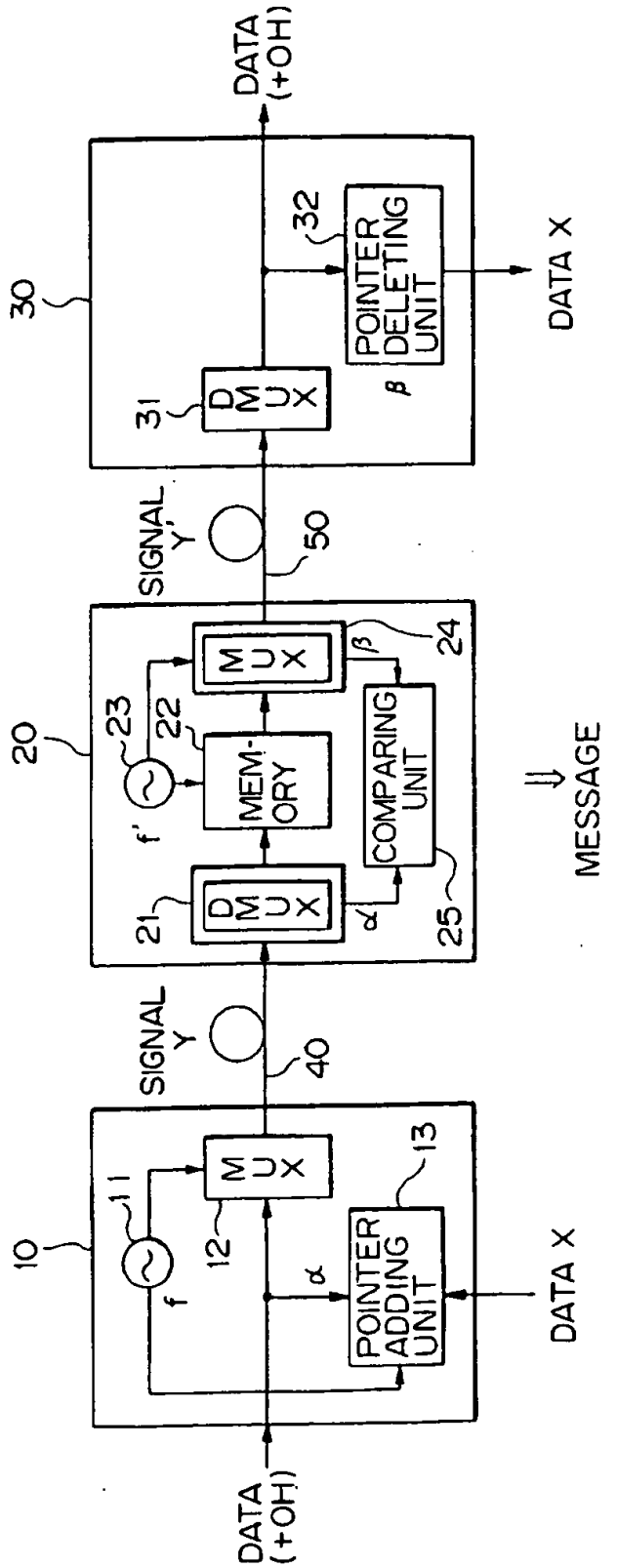

DATA (+OH)
30
32
POINTER DELETING UNIT
DATA X
31
DMUX
β
SIGNAL Y'
50
20
MUX
24
23
22
MEM-ORY
COMPARING UNIT
MESSAGE
f'
21
DMUX
α
25
SIGNAL Y
40
10
MUX
11
12
13
f
α
POINTER ADDING UNIT
DATA X
DATA (+OH)

Fig.7

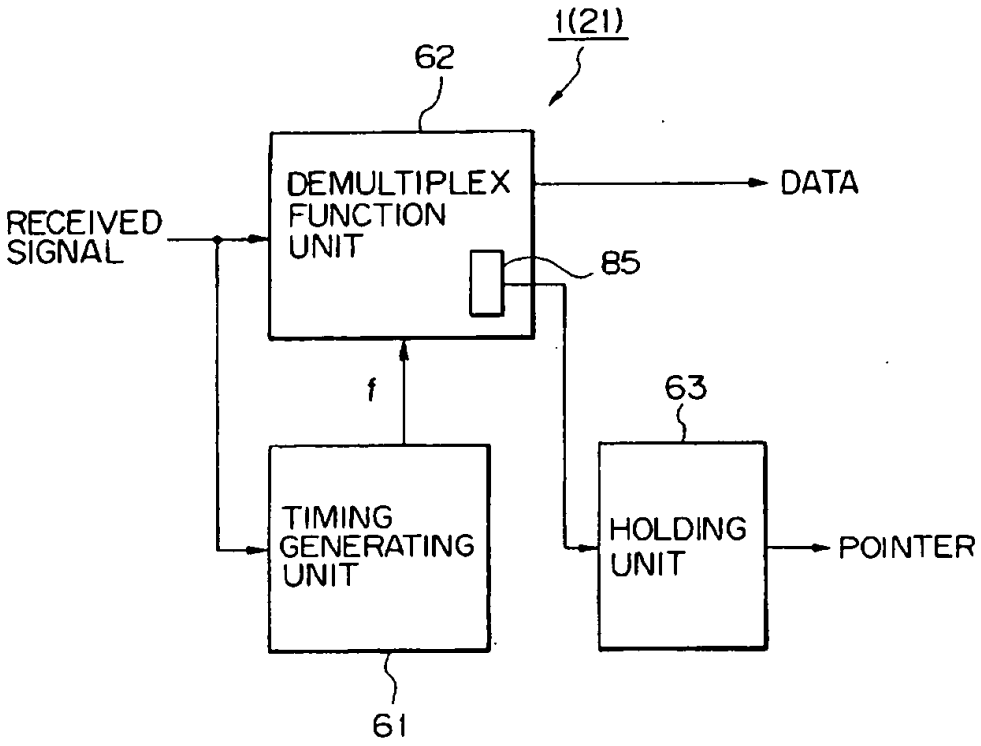

1(21)
62
DEMULTIPLEX FUNCTION UNIT
DATA
RECEIVED SIGNAL
85
f
63
TIMING GENERATING UNIT
HOLDING UNIT
POINTER
61

Fig.8

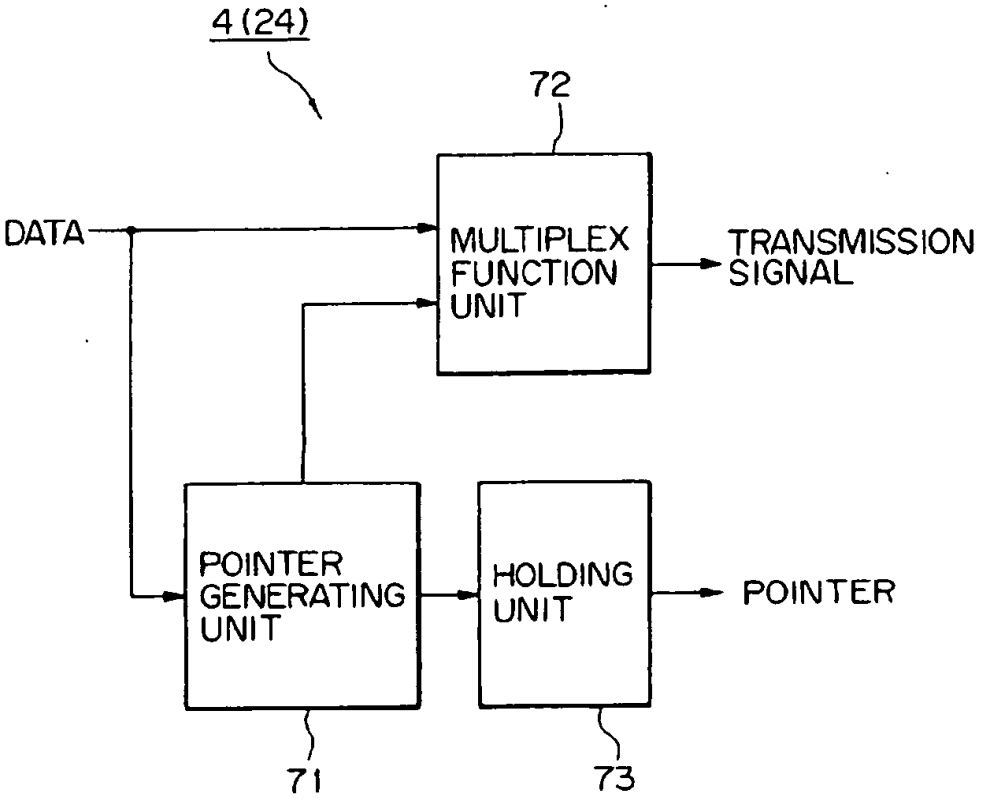

4(24)
72
DATA
MULTIPLEX FUNCTION UNIT
TRANSMISSION SIGNAL
POINTER GENERATING UNIT
HOLDING UNIT
POINTER
71
73

Fig. 9

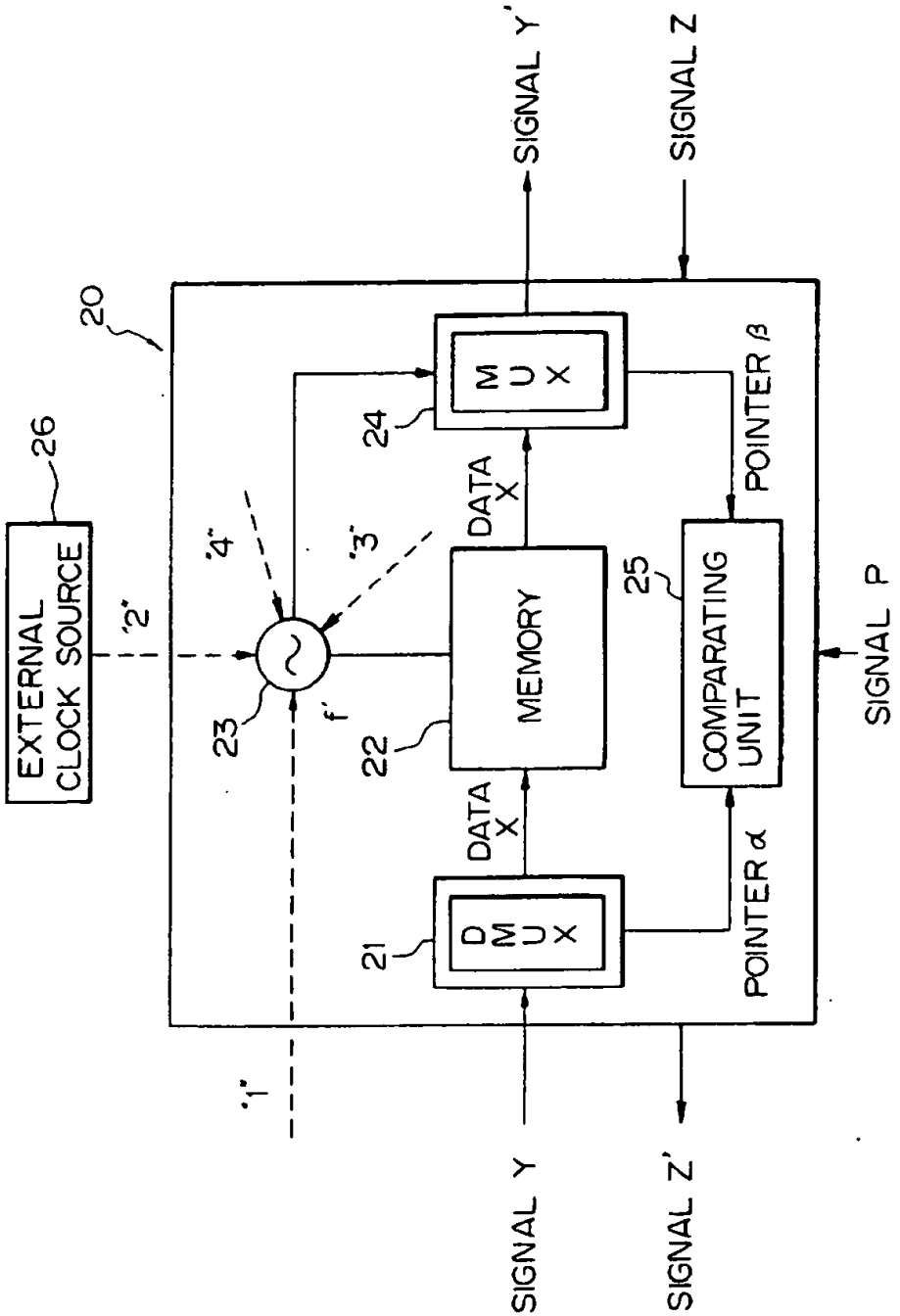
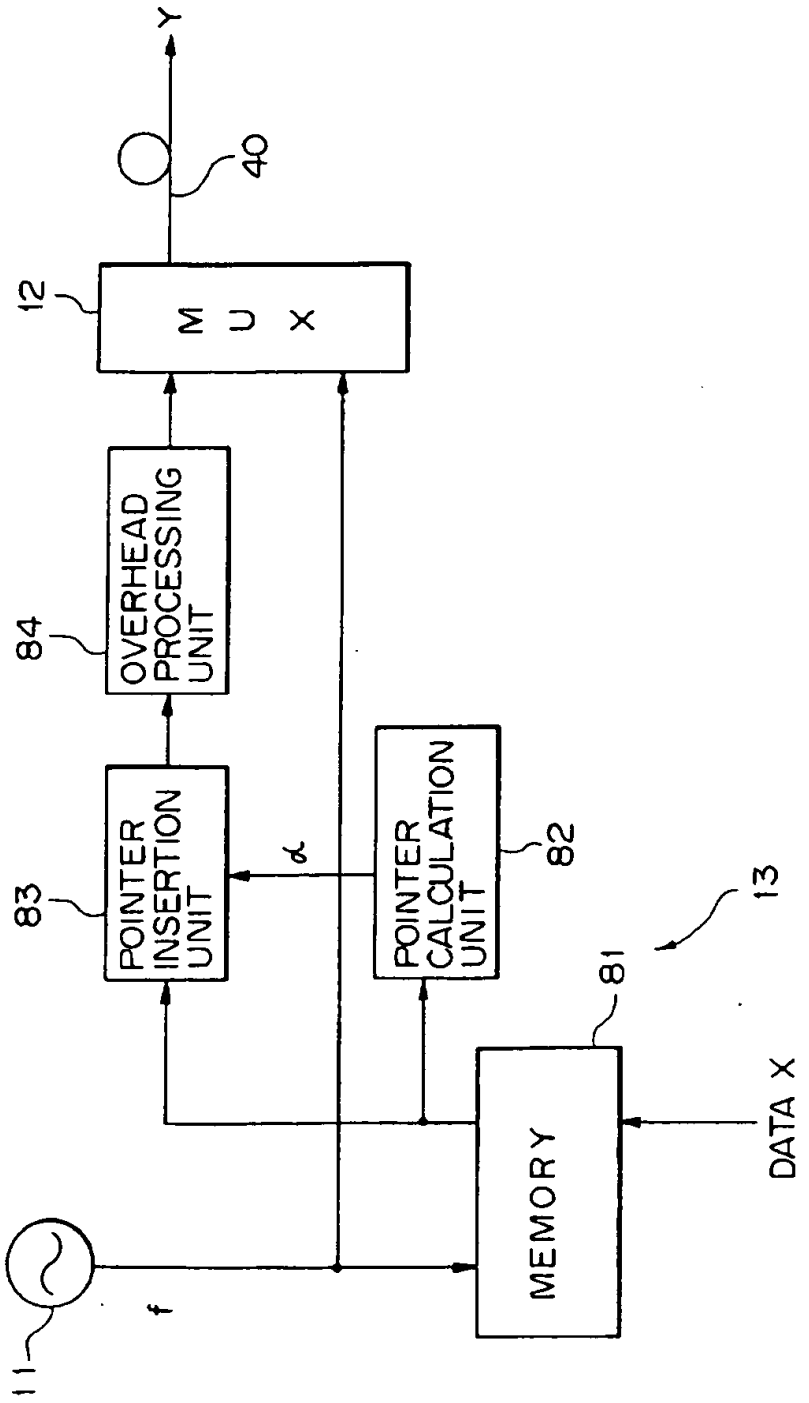

SIGNAL Y'
SIGNAL Z
20
M U X
24
POINTER β
DATA X
26
EXTERNAL CLOCK SOURCE
'4'
'3'
'2'
MEMORY
25
COMPARATING UNIT
SIGNAL P
23
f'
22
DATA X
D M U X
21
POINTER α
'1'
SIGNAL Y
SIGNAL Z'

Fig.10


10
Y
40
12
M U X
84
OVERHEAD PROCESSING UNIT
83
POINTER INSERTION UNIT
α
82
POINTER CALCULATION UNIT
13
81
MEMORY
DATA X
11
f

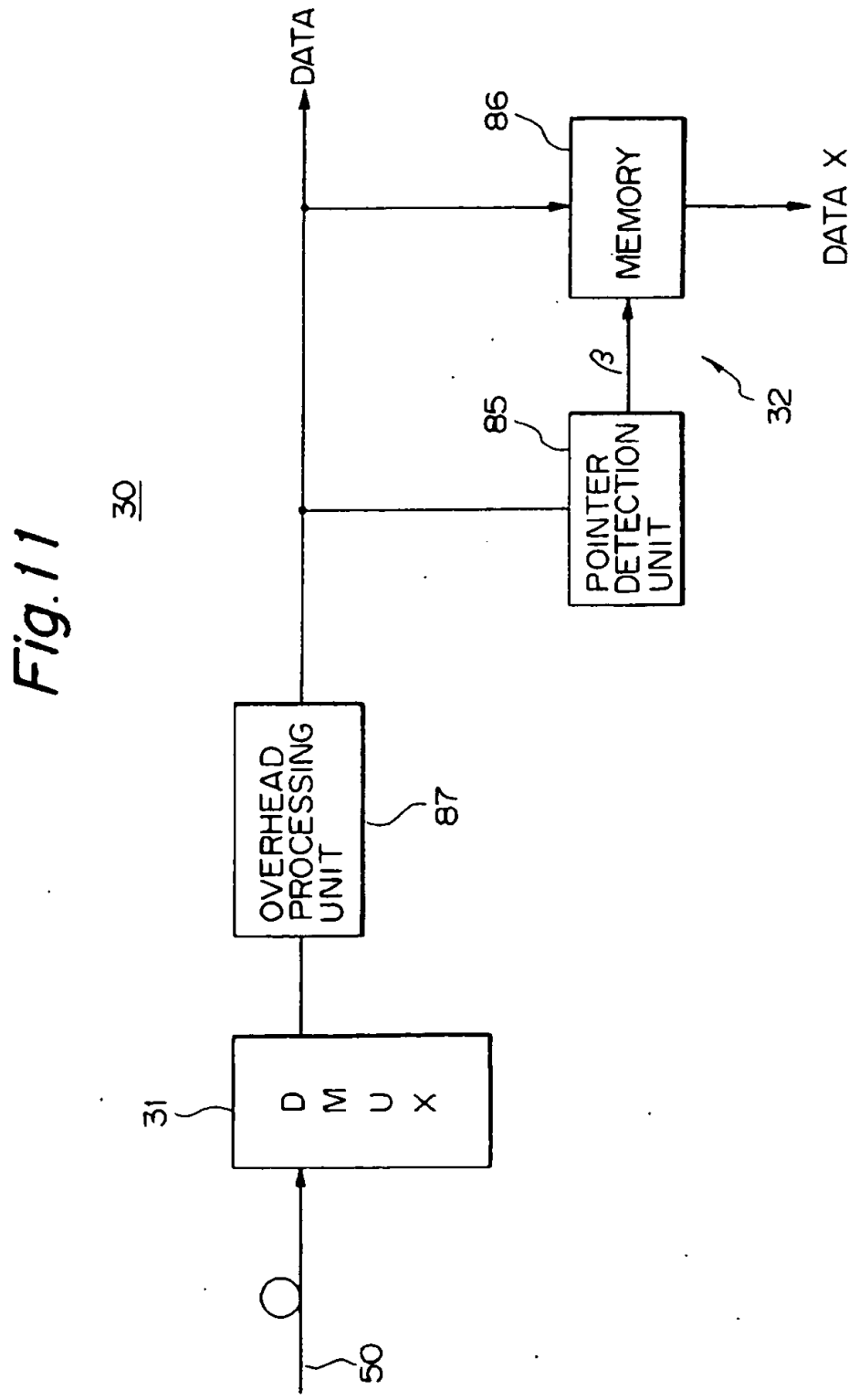
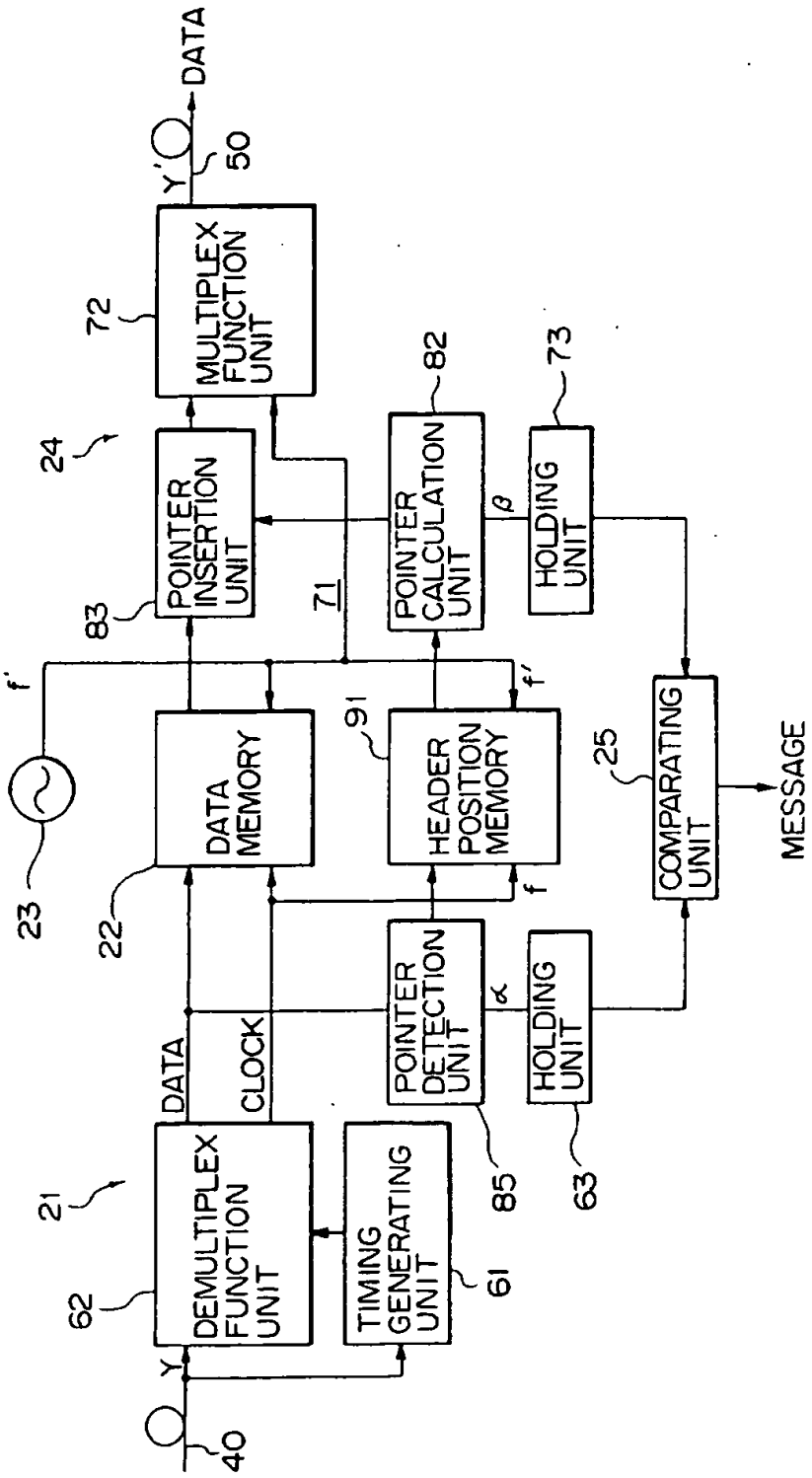

DATA
86
MEMORY
DATA X
β
32
85
POINTER DETECTION UNIT
30
OVERHEAD PROCESSING UNIT
87
D M U X
31
50

Fig. 11

Fig.12


DATA
Y'
50
MULTIPLEX FUNCTION UNIT
72
POINTER INSERTION UNIT
24
83
POINTER CALCULATION UNIT
82
HOLDING UNIT
73
β
71
f'
23
DATA MEMORY
22
HEADER POSITION MEMORY
91
f
COMPARATING UNIT
25
MESSAGE
POINTER DETECTION UNIT
85
α
HOLDING UNIT
63
DATA
CLOCK
DEMULTIPLEX FUNCTION UNIT
21
62
TIMING GENERATING UNIT
61
Y
40

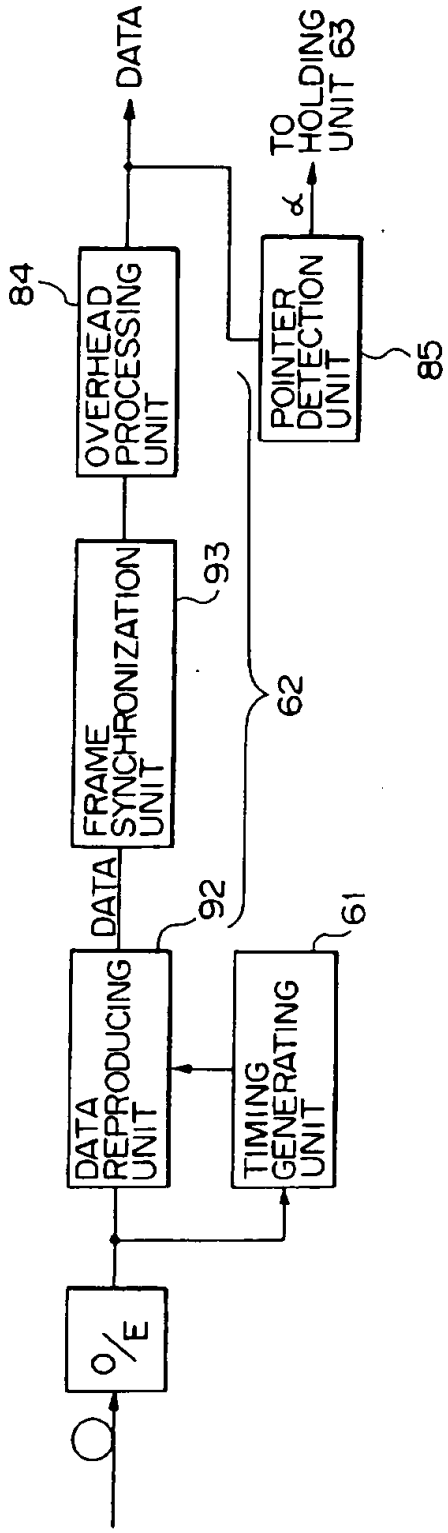

O/E
DATA REPRODUCING UNIT
92
DATA
FRAME SYNCHRONIZATION UNIT
93
OVERHEAD PROCESSING UNIT
84
DATA
TIMING GENERATING UNIT
61
62
POINTER DETECTION UNIT
85
TO HOLDING UNIT 63

Fig.13

NODE PROVIDED WITH FACILITY FOR CHECKING ESTABLISHMENT OF SYNCHRONIZATION

This is a continuation of application Ser. No. 08/213,254, filed Mar. 15, 1994 abandoned.

BACKGROUND OF THE INVENTION

1. Field of the Invention

The present invention relates to a system for checking for the establishment of synchronization in a synchronous communication network, more particularly relates to a system for checking for the establishment of synchronization which confirms the synchronization from the values of pointers of a received signal and transmission signal.

In recent years, along with the standardization of digital communication networks, synchronous multiplexing has spread even to the higher order region. As a result of this, it has become important to confirm the synchronization between transmission apparatuses in a synchronous communication network, such as SONET in the U.S.

2. Description of the Related Art

In conventional communication systems, a synchronous network is formed only at the lower order side. Since the speed is slow, asynchronization between transmission apparatuses does not pose that much of a problem. Therefore, there is no need to check the state of synchronization and accordingly there has been no continuous confirmation of synchronization performed.

Even in a synchronous communication network covered by the present invention, which will be explained later with reference to the drawings, there has been no general check of the establishment of synchronization including the higher order groups.

As explained above, if synchronous multiplexing is performed up to the higher order region due to the standardization of digital communication networks in recent years, asynchronization among transmission apparatuses sometimes causes trouble. Accordingly, the problem has arisen that it is not possible to check for the establishment of synchronization by a simple method or apparatus.

SUMMARY OF THE INVENTION

Accordingly, the present invention, in view of the above-mentioned problems, has as its object to check the establishment of synchronization by a simple means at each node.

To attain the above object, the present invention takes note of the values of the pointers added to the data of the signal received from the opposing node side and the values of the pointers added to the data when a signal is sent out from the home node as a transmission signal and judges that synchronization between the two nodes has been established when the values of the pointers match.

BRIEF DESCRIPTION OF THE DRAWINGS

The above object and features of the present invention will be more apparent from the following description of the preferred embodiments with reference to the accompanying drawings, wherein:

FIG. 1 is a view illustrating a multiplex hierarchy in a synchronous communication network;

FIG. 2 is a view of the overhead boundary in a synchronous communication network;

FIG. 3 is a view of mapping from VC-1 to VC-4;

FIG. 4 is a view of mapping from VC-4 to STM-1;

FIG. 5A is a view of the configuration of a node (transmission apparatus) in a synchronous communication network to which the principle of the present invention is applied;

FIG. 5B is a view of the signal formats of a received signal Y and transmission signal Y' in FIG. 5A;

FIG. 6 is a view of an embodiment of the present invention;

FIG. 7 is a view of an example of the configuration of a demultiplexer unit in the present invention;

FIG. 8 is a view of an example of the configuration of a multiplexer unit in the present invention;

FIG. 9 is a view of an example of the application of the present invention;

FIG. 10 is a view of a more specific example of the configuration of the node 10 in FIG. 6;

FIG. 11 is a view of a more specific example of the configuration of the node 30 in FIG. 6;

FIG. 12 is a view of a more specific example of the node 20 in FIG. 6; and

FIG. 13 is a view of an example of a demultiplex function unit shown in FIG. 7 and FIG. 12.

DESCRIPTION OF THE PREFERRED EMBODIMENTS

Before describing the embodiments of the present invention, the related art and the problems therein will be first described with reference to the related figures.

First, an explanation will be made of a known synchronous communication network to which the present invention is applied referring to FIG. 1 to FIG. 4.

FIG. 1 is a view illustrating a multiplex hierarchy in a synchronous communication network.

In FIG. 1, VC-12 is a basic virtual container, TU-12 is a tributary unit, TUG-21 and TUG-32 are tributary unit groups, VC-4 is a higher order virtual container, AU-4 is an administrative unit, and STM-1 is a synchronous transfer module.

The tributary unit TU-12 is formed by adding to the basic virtual container VC-12, formed from a 2.048 Mb/s container C-12, a path overhead VC-12 POH for transmitting control information among transmission apparatuses. By adding to the tributary unit TU-12 a pointer TU-12 PTR and multiplexing these in triplicate, the tributary unit group TUG-21 is formed.

By multiplexing seven tributary unit groups TUG-21, the tributary unit group TUG-32 is formed. Further, by multiplexing the tributary unit groups TUG-32 in triplicate and adding the path overhead VC-4 POH, the higher order virtual container VC-4 is formed.

By adding to the higher order virtual container VC-4 the path overhead VC-4 POH, the administrative unit AU-4 is formed. By adding to the administrative unit AU-4 the pointer AU-4 PTR, the synchronous transfer module STM-1 is formed.

The above multiplex hierarchy is a basic item in the CCITT Recommendations.

FIG. 2 is a view of the overhead boundary in a synchronous communication network. In FIG. 2, a path overhead VC-12 POH for transmitting control information between transmission apparatuses is added between the basic virtual containers VC-12, a path overhead VC-4 POH for transmit-

ting control information between transmission apparatuses is added between the higher order virtual containers VC-4, and a section overhead STM-N MSOH for transmitting control information between transmission apparatuses and a section overhead STM-N RSOH for transmitting control information between repeaters are added between the synchronous transfer modules STM-N.

The N in the above-mentioned "STM-N" stands for 1, 2, 3, etc. In FIG. 2, the example is shown where N=1. Further, MSOH means a multiplexer SOH, while RSOH means a regenerator SOH. Note that RSOH is, for example, an order-wire signal which can be monitored by the repeaters in the figure. MSOH cannot be monitored by these repeaters and can only be monitored by the STMs at the two ends. Incidentally, the signals C-12 are low order level signals used at the subscriber side.

FIG. 3 is a view of mapping from VC-1 to VC-4. In FIG. 3, the tributary unit group TUG 21 is formed by adding to a tributary unit 12, formed by adding to a basic virtual container VC-12 formed from a 2.048 Mb/s container C-12 a path overhead POH, V1 as the pointer TU-12 PTR and multiplexing these in triplicate.

By multiplexing seven tributary unit groups TUG-21, the tributary unit group TUG-32 is formed. Further, by multiplexing the tributary unit groups TUG-32 in triplicate and adding the path overhead VC-4 POH, the higher order virtual container VC-4 is formed.

FIG. 4 is a view of mapping from VC-4 to STM-1. In FIG. 4, an administrative unit AU-4 is formed by adding to the higher order virtual container VC-4 a pointer AU-4 PTR and a synchronous transfer module STM-1 is formed by adding to the administrative unit AU-4 the section overheads STM-1 RSOH and MSOH.

The above-mentioned AU-4 PTR is a value indicating the position of the so-called J1 byte in the figure. Using this, it is possible to specify the header position of data. When mapping data in the higher order virtual container VC-4, it is not set where the header position of the data will be. Therefore, a pointer is used to show the header position.

As explained above, synchronous multiplexing has been performed up to the higher order region due to the standardization of digital communication networks in recent years. As a result, asynchronization among transmission apparatuses sometimes causes trouble. Accordingly, it has become necessary to check for the establishment of synchronization by simple methods or apparatuses. Below, an explanation will be given of the present invention which enables the establishment of synchronization to be checked by a simple method.

FIG. 5A is a view of the configuration of a node (transmission apparatus) in a synchronous communication network to which the principle of the present invention is applied. FIG. 5B is a view of the signal formats of a received signal Y and transmission signal Y' in FIG. 5A.

The method of the present invention checks the establishment of synchronization at a node of a synchronous communication network which communicates by sending and receiving frames formed by successively adding pointers (α, β) to a plurality of data X to show their header positions. It compares the values α of the pointers added to the data X received from an opposing node (not shown in FIG. 5A, but at the left side of the figure) side and the values β of the pointers to be added to the data X transmitted from the home node (node shown in the figure) and, when detecting that the two values (α, β) match, recognizes that synchronization has been established between the opposing node and the home node.

More specifically, the method of the present invention

a) demultiplexes the signal Y received from the opposing node into the data X and pointers (α) by the timing of the received signal Y and holds the data X,

b) successively reads out the held data X by the clock (f') of the clock source 3 in the home node, adds pointers (α) showing the headers of the same to the data X, and multiplexes them to form the transmission signal Y',

c) compares the pointers (α) demultiplexed from the received signal Y and the pointers (β) added to the transmission signal Y', and

d) detects that the clock of the transmission node of the received signal Y and the clock of the home node are synchronized when the two pointers are compared and found to match.

Reference numeral 1 is a demultiplexer unit (DMUX) which takes out the data X from the received signal Y and monitors the values α of the pointers added to the data X at that time. Reference numeral 2 is a memory which holds the data X demultiplexed at the demultiplexer unit 1. Reference numeral 3 is a clock source which supplies a master clock (f') of the node (transmission apparatus). Reference numeral 4 is a multiplexer unit (MUX) which multiplexes the data X which has been demultiplexed and forms the transmission signal Y' and also, at that time, confirms the headers of the data X, adds pointers (β) showing the positions of the same, and continuously monitors the values β of the pointers at that time. Reference numeral 5 is a comparing unit which compares the values α of the pointers monitored at the demultiplexer unit 1 and the values β of the pointers monitored at the multiplexer unit 4.

The data X demultiplexed from the received signal Y at the demultiplexer unit 1 are written into the memory 2 as they are in synchronization with the speed of the received signal. At this time, the demultiplexer unit 1 monitors the values α of the pointers added to the data X.

In the example shown in FIG. 1 to FIG. 4, since the 63 tributary units mapped in a fixed order in the synchronous transfer module STM-1 have been given V1 as pointer values α, this is monitored for each tributary unit.

The data X written into the memory 2 are read out from the memory 2 in synchronization with the master clock (f') of the node (transmission apparatus) from the clock source 3 and are multiplexed into the signal Y' at the multiplexer unit 4. At this time, the multiplexer unit 4 confirms the position of the headers of the data X, adds pointer values β showing those positions, and monitors the values.

In the example shown in FIG. 1 to FIG. 4, the multiplexer unit 4 successively reads out 63 tributary units from the memory 2 in the same order as at the demultiplexer unit 1 side and once again composes the synchronous transfer module STM-1. At this time, it adds V1 as the pointer values β in the same order.

The comparing unit 5 compares the pointer values α being monitored in the demultiplexer unit 1 and the pointer values β being monitored in the multiplexer unit 4 and, if α and β match, it is judged that synchronization has been established between the node (transmission apparatus) and the apparatus of the opposing node transmitting the received signal Y. Further, when α and β do not match, it is judged that there is a difference in the speed of the clocks from the clock sources used as master clocks by the transmission apparatuses and therefore synchronization is not established.

This processing is performed on the pointer values α and β with respect to the header positions $A_1, A_2, \ldots$ of the data

$X_1, X_2, \ldots$ in the received signal Y and the header positions $B_1, B_2, \ldots$ of the data $X_1, X_2, \ldots$ in the transmission signal Y' and the coincidence of the results of comparison of α and β is watched so as to check the establishment of synchronization (between the opposing node and home node). These $X_1, X_2, \ldots, A_1, A_2, \ldots, B_1, B_2, \ldots$ are as shown in FIG. 5B. Note that the pointer shown in FIG. 5B corresponds, for example, to the pointer AU-4 PTR in FIG. 4. The RSOH and MSOH adjoining this AU-4 PTR are omitted from the illustration in FIG. 5B.

In this case, the pointers are compared between the overheads of the received signal and transmission signal given path overheads or section overheads of the same level. For example, referring to FIG. 2, a comparison is made by the comparing unit 5 between the pointers belonging to the path overhead VC-12 POH level.

FIG. 6 is a view of an embodiment of the present invention. Reference numerals 10, 20, and 30 show nodes (offices) in a synchronous communication network. It shows only the important parts of the configuration at the nodes when transmitting data X from the node 10 through the node 20 to the node 30. Further, 40 shows a transmission line connecting the node 10 and the node 20, while 50 shows a transmission line connecting the node 20 and the node 30.

At the opposing node 10, reference numeral 11 is a clock source which supplies a master clock (f) at the node 10. Reference numeral 12 is a multiplexer unit (MUX) which multiplexes the data X to form a transmission signal Y. Reference numeral 13 is a pointer adding unit which adds the pointer values α to the data X.

At the home node 20 (corresponding to FIG. 5A), 21 is a demultiplexer unit (DMUX) which receives the signal Y from the node 20 and demultiplexes the data X and the pointers (α). Reference numeral 23 is a clock source which supplies the master clock (f') at the node 20. Reference numeral 22 is a memory which holds the data X demultiplexed at the demultiplexer unit 21. Reference numeral 24 is a multiplexer unit (MUX) which remaps the data X read out from the memory 2 in synchronization with the clock (f') and adds the pointer values β to the data X to form the transmission signal Y'. Reference numeral 25 is a comparing unit which compares the pointer values α and the pointer values β.

At the node (downstream side) 30, 31 is a demultiplexer unit (DMUX) which receives the signal Y' and demultiplexes it to the data X and the pointers (β). Reference numeral 32 is a pointer deleting unit which removes the demultiplexed pointers and outputs the data X.

At the opposing node 10, the data X are multiplexed to the signal Y (mapped into Y) in synchronization with the master clock (f). At this time, the pointer values α showing the headers of the data X in the signal Y are added. The signal Y sent from the node 10 is transmitted through the transmission line 40 and received at the home node 20.

At the home node 20, the data X is taken out from the signal Y and held in the memory 23 and the values α of the pointers which have been added are monitored. Further, the data X are multiplexed by remapping into the signal Y' in synchronization with the master clock (f') of the node 20. At this time, the values α of the pointers and the values β of the pointers are compared and if α does not equal β, it is judged that the clock f does not equal f'. By this, it is confirmed that the clocks of the node 10 and the node 20 are not synchronized. The signal Y' sent from the node 20 is sent through the transmission line 50 and received at the downstream node 30.

At the node 30, the data X are taken out from the signal Y', the added pointers (β) are demultiplexed and removed by the pointer deleting unit 32, and thus the data X are extracted.

Note that in FIG. 6, the nodes 10, 20, and 30 are similar to each other in configuration, but only the portions required for explaining the operations at the nodes are drawn. As mentioned above, the node 20 of FIG. 6 corresponds to the node shown in FIG. 5A, and the members 1, 2, 3, 4, and 5 in FIG. 5A correspond to the members 21, 22, 23, 24, and 25 in FIG. 6.

In this way, according to the present invention, by detecting the coincidence of the pointers of the received signal and the transmission signal, it is possible to confirm the state of synchronization between nodes (transmission apparatuses). This information confirming synchronization can be used as alarm information for synchronization between transmission apparatuses. For example, it may be used for transmitting a message showing that they are out of synchronization.

FIG. 7 is a view of an example of the configuration of a demultiplexer unit in the present invention. Reference numeral 61 is a timing generating unit which generates a timing signal from the received signal, 62 is a demultiplex function unit which performs a demultiplex function for demultiplexing the received signal into the data and pointers, and 63 is a holding unit for holding the demultiplexed pointers.

At the timing generating unit 61, the timing component is extracted from the received signal to generate the timing signal. At the demultiplex function unit 62, the timing signal produced by the timing generating unit 61 is used to demultiplex the received signal into the data portion and the pointers showing the headers of the data. The data are then output and the demultiplexed pointers held temporarily in the holding unit 63.

FIG. 8 is a view of an example of the configuration of a multiplexer unit in the present invention. In the figure, reference numeral 71 is a pointer generating unit for generating pointers to be added to data read out from the memory, 72 is a multiplex function unit which functions to multiplex the data and the pointers and produce a transmission signal, and 73 is a holding unit for holding the pointers produced in the pointer generating unit.

The pointer generating unit 71 produces pointers corresponding to the data successively read out from the memory. At the multiplex function unit 72, the corresponding pointers are added to the data successively read out from the memory so as to form frames and produce transmission signals and the added pointers are temporarily held in the holding unit 73.

FIG. 9 is a view of an example of the application of the present invention. This figure illustrates the structure of the home node 20 shown in FIG. 6. Reference numeral 26 is an external clock source provided at the node 20, the signals Z and Z' are downstream signals when the signals Y and Y' at the node 20 are made the upstream signals, for example, and P is a signal from a subnode (tributary) merging at the node 20. Further, "1" is the timing signal extracted from the received signal Y, "2" is the master clock from the external clock source 26, "3" is the timing signal extracted from the signal P from the subnode, and "4" is the timing signal extracted from the downstream signal Z.

Assuming now that the timing signal "1" from the received signal Y is used as the master clock source (f'), since the pointers α are always equal to β, there is synchronization with the opposing node sending out the signal Y.

However, when the timing signal "4" is used as the master clock source (f'), when synchronization is not established in the communication network as a whole due to some reason or another, a frequency difference is created between the clock of the signal Y clock source and the master clock (f') at the node 20.

Therefore, a plurality of clock sources are provided at the home node. These plurality of clock sources are periodically successively selected and used. The one clock source giving the result with the least noncoincidence in the comparison is finally selected and used as the representative clock source for the home node.

That is, the pointer values in the case of use of the values of the plurality of clock sources able to be periodically selected by the transmission apparatuses are compared and the single clock source giving the smallest frequency difference is selected as the master clock (f') of the apparatus. In this way, by making use of the pointer values to check the quality of the clock source, it is possible to continuously establish a synchronous network.

Note that in this case, if use is made of the timing signal "1" at all times, it might be thought that no problem would arise, but when there is some problem such as an abnormality in the apparatus at the opposing node side of the signal Y, it is not possible to use the timing signal "1". By setting up the system as in the present example of application, it is possible to select the more accurate clock source at all times.

Finally, more specific examples of the configuration of several of the constituent elements shown in FIG. 6, FIG. 7, and FIG. 8 will be shown.

FIG. 10 is a view of a more specific example of the configuration of the node 10 in FIG. 6. In particular, it shows more specifically the pointer adding unit 13 of FIG. 6. As illustrated, the pointer adding unit 13 is comprised of a memory 81 for holding temporarily the data X, a pointer calculation unit 82 for calculating (counting) where the headers of the data X are so as to calculate the pointer values β, and a pointer insertion unit 83 for writing the values β in pointer regions. After this, it is connected through an overhead processing unit 84 to a multiplexer unit 12.

The format of the signal Y is formed in synchronization with the master clock (f) and is sent to the node 20, but in this case a fixed value is always used as the values α of the pointers to be inserted. The reason is that the data X from the memory 81 is also synchronized with the clock (f).

FIG. 11 is a view of a more specific example of the configuration of the node 30 in FIG. 6. In particular, it shows more specifically the pointer deleting unit 32 in FIG. 6.

The signal output from the demultiplexer unit 31 passes through the overhead processing unit 87, then the pointer detection unit 85 detects its pointer portions. Further, the values β written into the pointers are read out.

On the other hand, the above-mentioned signal is stored temporarily in the memory 86 as well. Only the data X to be dropped at the node 30 are read out from the memory 86 by accessing by the pointer values β. The portions accessed by β are the header positions of the data X.

FIG. 12 is a view of a more specific example of the node 20 in FIG. 6. Note that constituent elements substantially the same as those already explained are given the same reference numerals.

The demultiplexer unit 21 of FIG. 6 is comprised of a demultiplex function unit 62 (FIG. 7), a timing generating unit 61 (FIG. 7), a pointer detection unit 85 (FIG. 11), and a holding unit 63 (FIG. 7). Note that in this figure, the detection unit 85 is drawn pulled out from the function unit 62.

The multiplexer unit 24 of FIG. 6 is comprised of a pointer calculation unit 82 (FIG. 10) and pointer insertion unit 83 (FIG. 10) forming the pointer generating unit 71 shown in FIG. 8, a multiplex function unit 72 (FIG. 8), and a holding unit 73 (FIG. 8).

In a specific example of the present invention, provision is made of a header position memory 91 between the pointer detection unit 85 and pointer calculation unit 82. Each time the pointer detection unit 85 detects a pointer, the detection is written, in a form of pulse, into the memory 91 in a time series. The timing at this time is the clock (f).

On the other hand, the detection pulse stored in the memory 91 is read out in time series in synchronization with the clock (f'). In synchronization with this readout operation, the header positions of the data read out from the memory 22 are calculated (counted) to find the values β of the pointers.

FIG. 13 is a view of an example of a demultiplex function unit shown in FIG. 7 and FIG. 12. Constituent elements substantially the same as those already explained are given the same reference numerals. In the demultiplex function unit 62, the data reproducing unit 92 and the frame synchronization unit 93 for detecting the frame byte (header) of the received data are constituent elements shown for the first time in this figure. Note that the pointer detection unit 85 operates to detect pointer areas (PTR) located at the portions separated by a certain predetermined number of bytes from the frame byte (header) detected by the unit 93.

As explained above, the present invention enables continuous confirmation of the synchronization among transmission apparatuses in a synchronous communication network and therefore contributes greatly to the improvement of the reliability of communication networks overall.

We claim:

1. A method for checking the establishment of synchronization at a node in a synchronous communication network which performs communication by sending and receiving multiplexed frames, comprising the steps of:

forming said frames by successively adding pointers showing header positions to a plurality of data portions;

comparing at said node between the value of a pointer included in each data portion received from an opposing node side and the value of a pointer to be added to each data portion, which is the same as the received data portion, and to be sent from a home node;

detecting whether the compared values coincide, synchronization being established between the opposing node and the home node when said compared values coincide;

demultiplexing each of said frames received at said home node in a received signal from the opposing node into said data portions and said pointers in correspondence with a timing component in the received frames;

holding the data in a memory device,

successively reading out the held data portions in correspondence with a clock of a clock source in the home node, adding pointers to the read out data portions at the home node showing headers of the data portions to be sent and multiplexing to form frames of a transmission signal from said home node,

comparing the pointers demultiplexed from the received frames and the pointers added to the transmission signal, and

detecting that a clock of the opposing node that provided the received signal and the clock of the home node are synchronized when the compared pointers are found to coincide.

wherein a plurality of clock sources are provided at the home node and

a further step of periodically successively selecting and using the plurality of clock sources, one of said plurality of clock sources giving a result with least non-coincidence in the comparison being finally selected and used as a representative clock source for the home node.

2. An apparatus for checking for the establishment of synchronization at a node in a synchronous communication network which performs communication by sending signals and receiving signals of multiplexed frames formed by successively adding pointers showing header positions to a plurality of data portions, comprising:

a demultiplexer unit (1) at a home node for demultiplexing a received signal sent from a transmission node into data portions and pointers in correspondence with a timing of the received signal,

a memory (2) for holding the demultiplexed data portions,

a multiplexer unit (4) for successively reading out the held data from said memory in time with a clock of a clock source (3) of the home node, for adding pointers showing headers of the read out data and for multiplexing said read-out data and added pointers to prepare a transmission signal to be sent from said home node, and

a comparing unit (5) for comparing the pointers separated from said received signal by the demultiplexer unit (1) and the pointers added to said transmission signal by the multiplexer unit (4), so as to find whether or not the synchronization between a clock of the transmission node of the received signal and the clock of the home node is established by the resultant comparison and, if it is not established, said clock of said home node is adjusted without adjusting the pointers.

3. An apparatus for checking for the establishment of synchronization at a node in a synchronous communication network which performs communication by sending signals and receiving signals of multiplexed frames formed by successively adding pointers showing header positions to a plurality of data portions, comprising:

a demultiplexer unit (1) at a home node for demultiplexing a received signal sent from a transmission node into data portions and pointers in correspondence with a timing component in the received signal,

a memory (2) for holding the demultiplexed data portions,

a multiplexer unit (4) for successively reading out the held data from said memory in time with a clock of a clock source (3) of the home node, for adding pointers showing headers of the read out data and for multiplexing said read-out data and added pointers to prepare a transmission signal to be sent from said home node, and

a comparing unit (5) for comparing the pointers separated from said received signal by the demultiplexer unit (1) and the pointers added to said transmission signal by the multiplexer unit (4),

synchronization between a clock of the transmission node of the received signal and the clock of the home node being detected by coincidence in the comparison,

wherein said demultiplexer unit (1) includes a demultiplex function unit for said demultiplexing of the received signal into said data portions and pointers, and a first holding unit (63) for holding the demultiplexed pointers,

said multiplexer unit (4) includes a multiplex function unit (72) for successively reading out the data portions held in the memory (2), timed by the clock of the home node, and for successively adding said pointers to said read-out data portions indicating the respective headers of the read-out data portions and multiplexing said read-out data portions and pointers to prepare said transmission signal, and includes a second holding unit for holding the added pointers, and

said comparing unit (5) detecting the coincidence of the pointers held in the first holding unit (63) and the pointers held in the second holding unit (73) coincidence of said pointers corresponding to synchronization between the clock of the transmission node of the received signal and the clock of the home node.

4. An apparatus for checking for the establishment of synchronization as set forth in claim 3, wherein said multiplexer unit (4) includes a pointer generating unit (71) for adding pointers to said read-out data portions.

5. An apparatus for checking for the establishment of synchronization as set forth in claim 4, further comprising:

a pointer detection unit (85) provided in said demultiplexer unit for detecting said received pointers in said received signal,

a pointer calculation unit (82) provided in said pointer generating unit (71) for adding said added pointers to said data portions read-out from memory, and

a header position memory, connected between said pointer detection unit and said pointer calculation unit for writing detection pulses by a first clock each time the pointer detection unit detects a pointer, reading out detection pulses by a second clock, and causing the calculation of the header position of the received data at the pointer calculation unit.

6. An apparatus for checking for the establishment of synchronization as set forth in claim 3, wherein said demultiplexer unit (1) includes a timing generating unit (61) for extracting the timing component of said received signal.

* * * * *

US005828670A

United States Patent [19]

Narasimha et al.

[11] Patent Number: 5,828,670

[45] Date of Patent: Oct. 27, 1998

[54] **DISTRIBUTION OF SYNCHRONIZATION IN A SYNCHRONOUS OPTICAL ENVIRONMENT**

[75] Inventors: **Madihally Narasimha; Kishan Shenoi**, both of Saratoga, Calif.

[73] Assignee: **Symmetricom, Inc.**, San Jose, Calif.

[21] Appl. No.: **467,313**

[22] Filed: **Jun. 6, 1995**

[51] **Int. Cl.**[6] **H04J 3/06**

[52] **U.S. Cl.** **370/516**; 370/507; 375/293; 375/326

[58] **Field of Search** 370/105.3, 105.2, 370/105.1, 102, 100.1, 17, 60, 103, 507, 516, 509, 505, 503, 252, 422; 375/293, 226, 356, 326

[56] **References Cited**

U.S. PATENT DOCUMENTS

4,270,203	5/1981	Collins et al.	370/102
4,489,421	12/1984	Burger	375/112
4,494,211	1/1985	Schwartz	375/356
4,720,829	1/1988	Fukasawa et al.	371/5.1
4,890,303	12/1989	Bader	375/107
4,912,706	3/1990	Eisenberg et al.	370/103
4,998,242	3/1991	Upp	370/60
5,172,376	12/1992	Chopping et al.	370/503
5,222,102	6/1993	Remson	375/293
5,241,543	8/1993	Amada et al.	370/100.1
5,450,394	9/1995	Gruber et al.	370/17
5,499,236	3/1996	Giallorinzi et al.	370/320
5,528,609	6/1996	Asano	370/516
5,555,261	9/1996	Nakayama et al.	370/103
5,604,771	2/1997	Quiros	375/326

FOREIGN PATENT DOCUMENTS

0460835	5/1991	European Pat. Off.	H04J 3/06

OTHER PUBLICATIONS

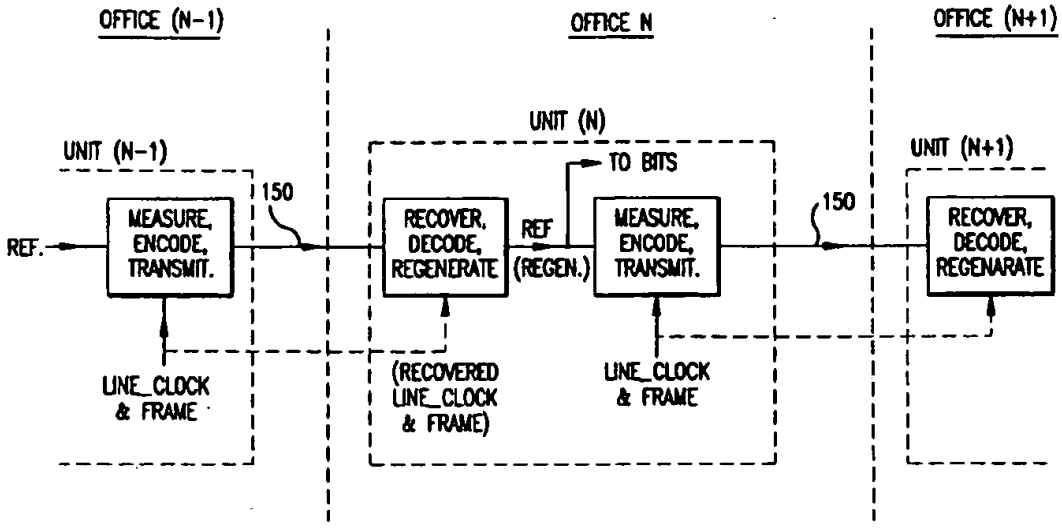
Lau, Richard C., and Fleischer, Paul E., Synchronous Techniques For Timing Recovery In BISDN Accepted for publication in IEEE *Transactions on Communications*, (Jul. 31, 1995).

Primary Examiner—Douglas W. Olms
Assistant Examiner—Shick Hom
Attorney, Agent, or Firm—Wilson, Sonsini, Goodrich & Rosati

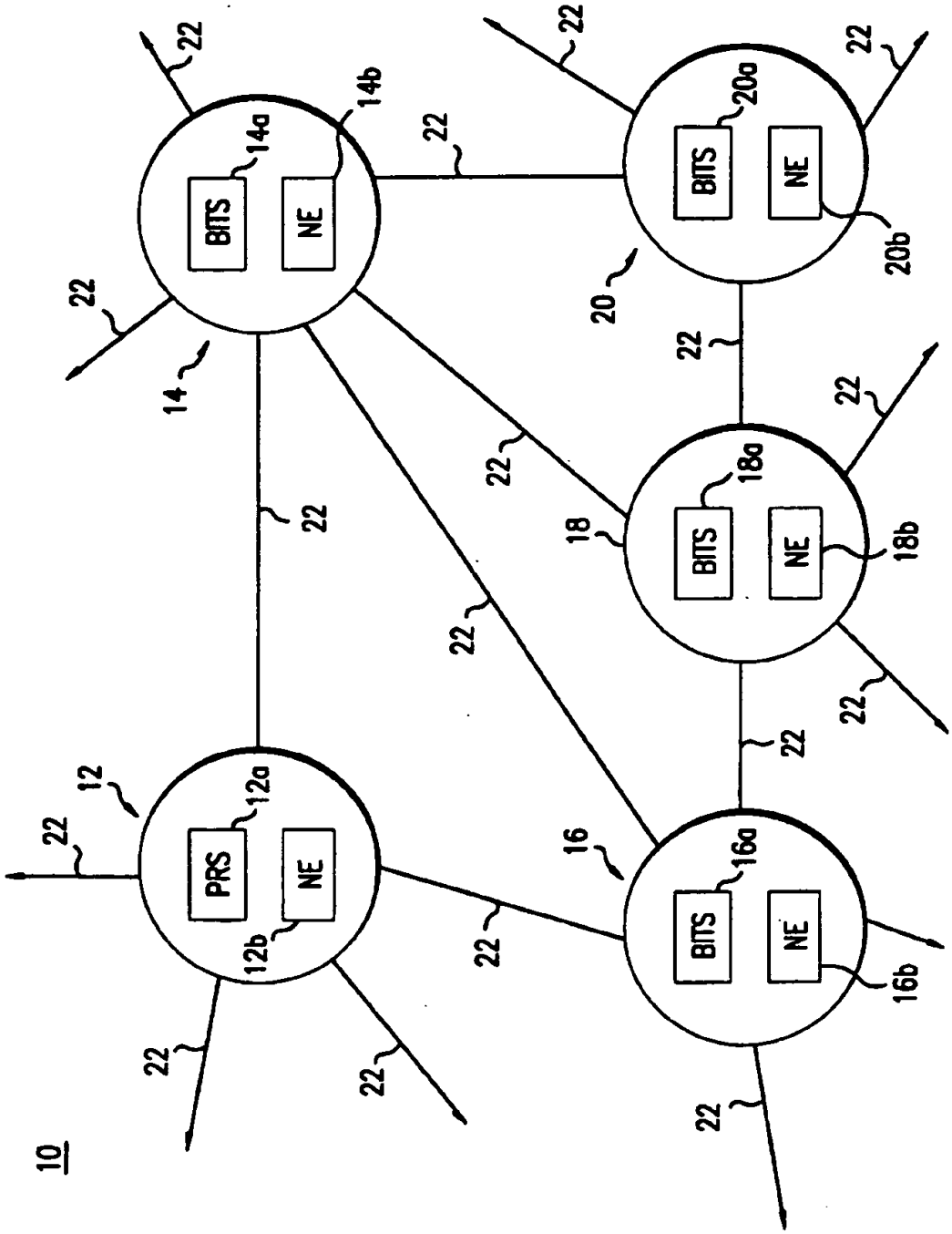
[57] **ABSTRACT**

Apparatus and methods for distributing synchronization throughout a network is disclosed. The distribution of the synchronization is through the use of generating a reference timing signal, and by counting the line clock pulses between the start of a frame and the timing reference signal pulse at a first office and that count is then encoded and transmitted to the next office. At the next office, the transmitted count is decoded and used for regenerating synchronization by counting a number of received line clock pulses from the start of the frame to regenerate the reference timing signal. Particular criteria for selecting the frequencies for the timing reference signal are disclosed.

33 Claims, 11 Drawing Sheets



Prior Art
FIG. 1



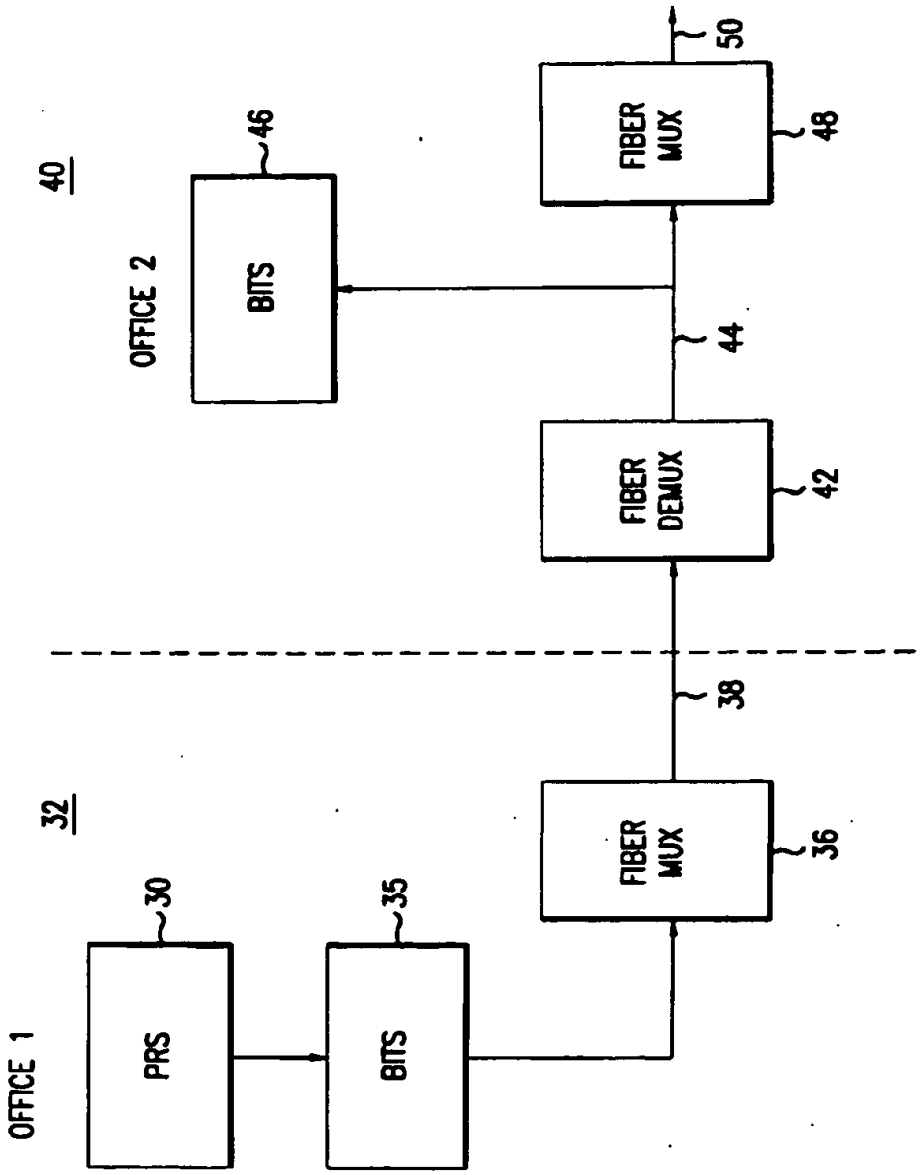


FIG.2 Prior Art

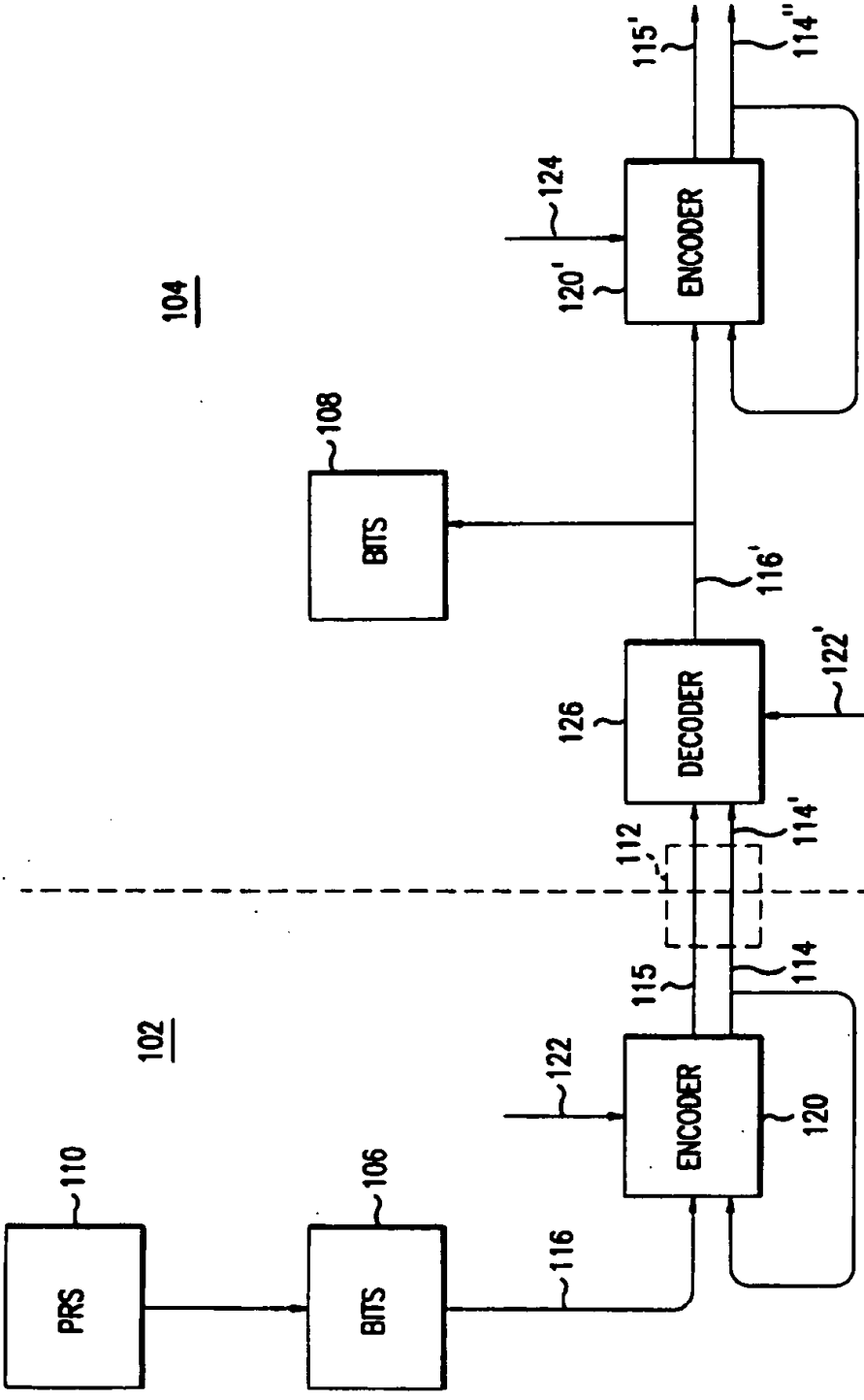

104
102
100
PRS
110
BITS
106
BITS
108
ENCODER
120
ENCODER
120'
DECODER
126
122
122'
124
115
115'
114
114'
114"
116
116'
112'

FIG.3

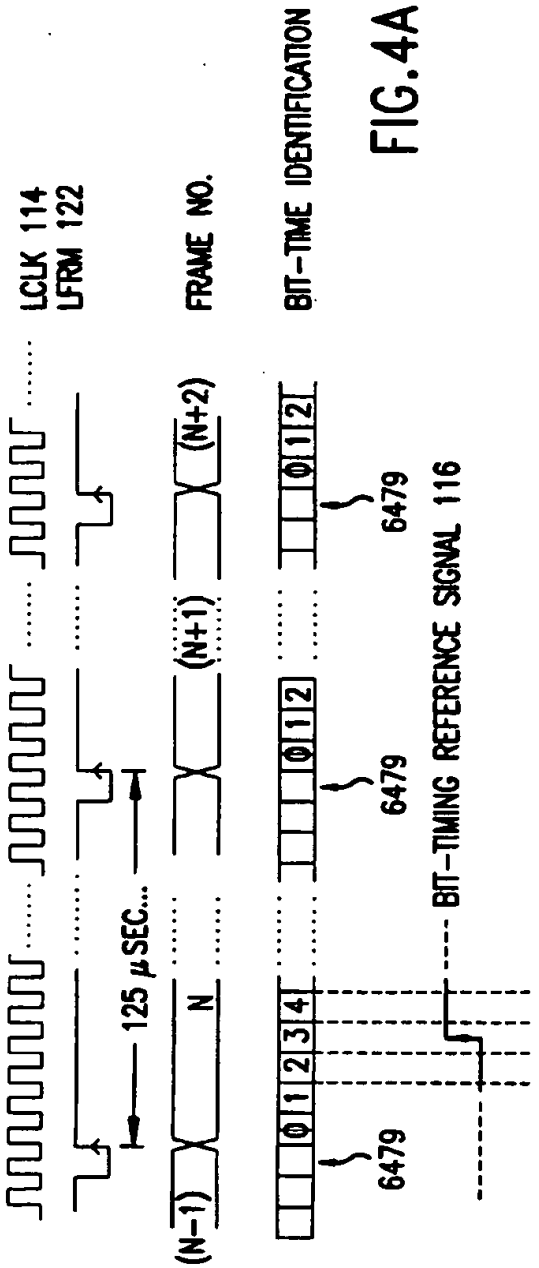

LCLK 114
LFRM 122
125 μSEC...
(N-1)
N
(N+1)
(N+2)
FRAME NO.
BIT-TIME IDENTIFICATION
0 1 2 3 4
0 1 2
6479
6479
BIT-TIMING REFERENCE SIGNAL 116

FIG.4A

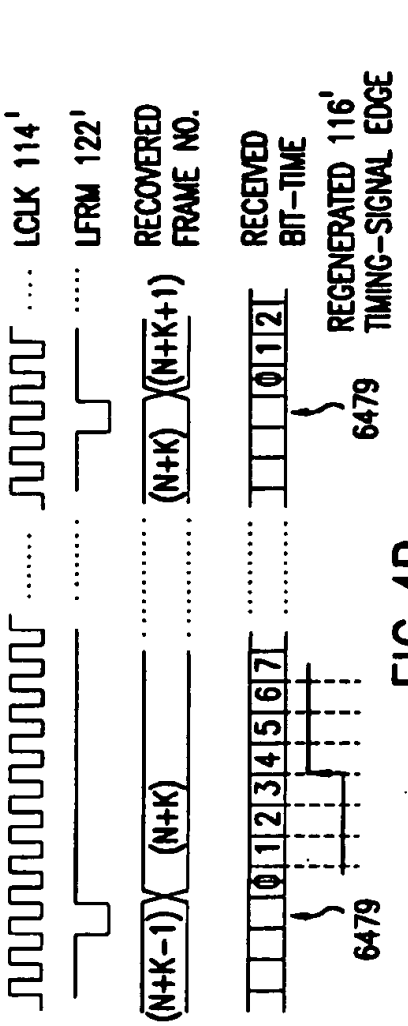

LCLK 114'
LFRM 122'
(N+K-1)
(N+K)
(N+K)
(N+K+1)
RECOVERED FRAME NO.
0 1 2 3 4 5 6 7
0 1 2
6479
6479
RECEIVED BIT-TIME
REGENERATED 116' TIMING-SIGNAL EDGE

FIG.4B

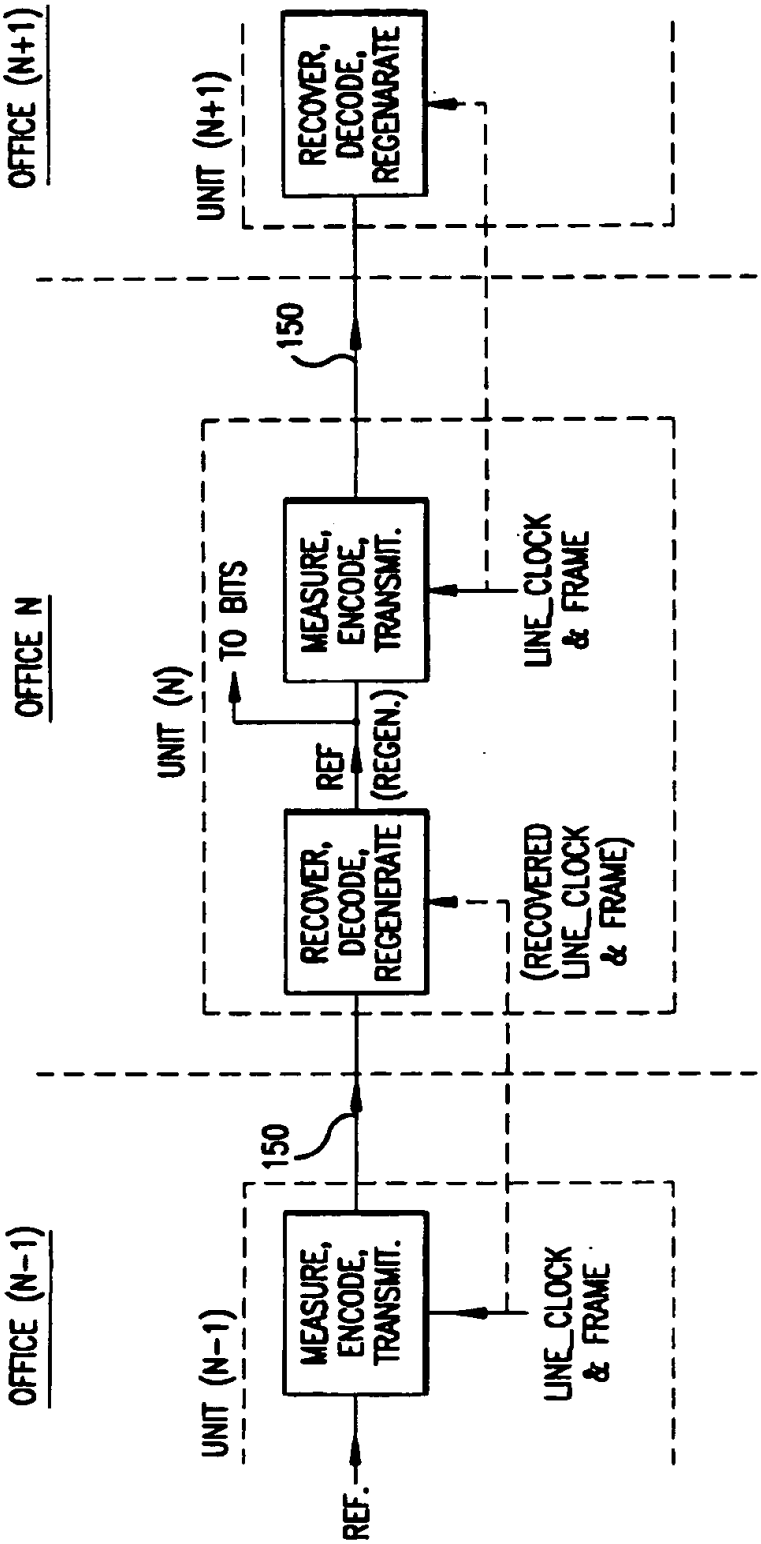

OFFICE (N+1)
UNIT (N+1)
RECOVER, DECODE, REGENARATE
150
OFFICE N
UNIT (N)
TO BITS
MEASURE, ENCODE, TRANSMIT.
LINE_CLOCK & FRAME
REF
(REGEN.)
RECOVER, DECODE, REGENERATE
(RECOVERED LINE_CLOCK & FRAME)
150
OFFICE (N-1)
UNIT (N-1)
MEASURE, ENCODE, TRANSMIT.
LINE_CLOCK & FRAME
REF.

FIG.5

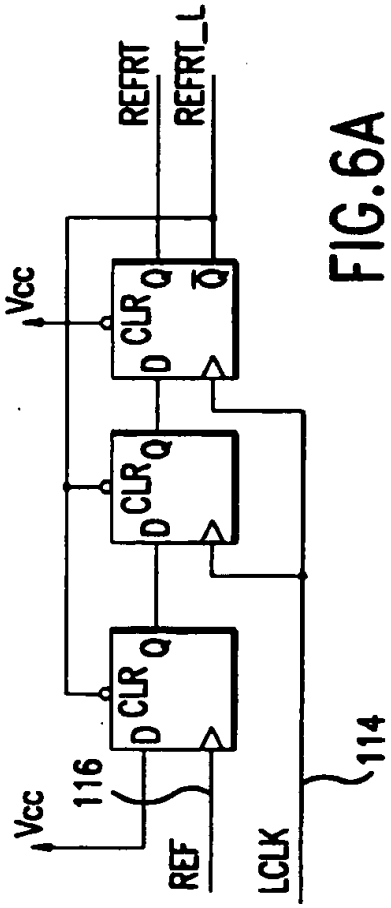

REFRT
REFRT_L
Vcc
CLR
D
Q
Q
Vcc
116
REF
LCLK
114

FIG.6A

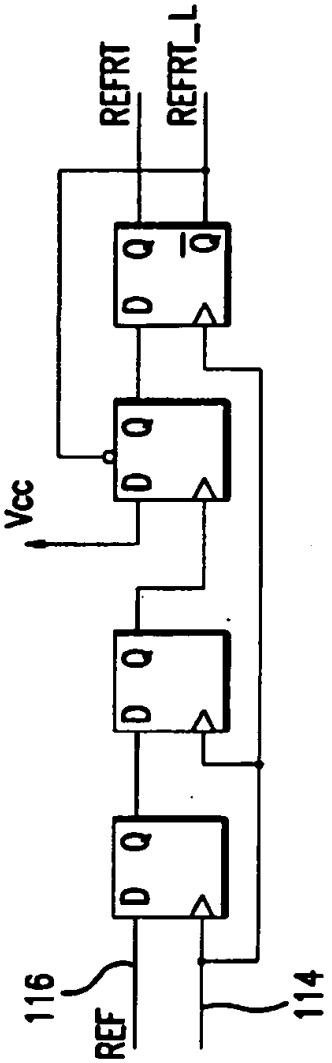

REFRT
REFRT_L
Q
Q
D
Vcc
116
REF
114

FIG.6B

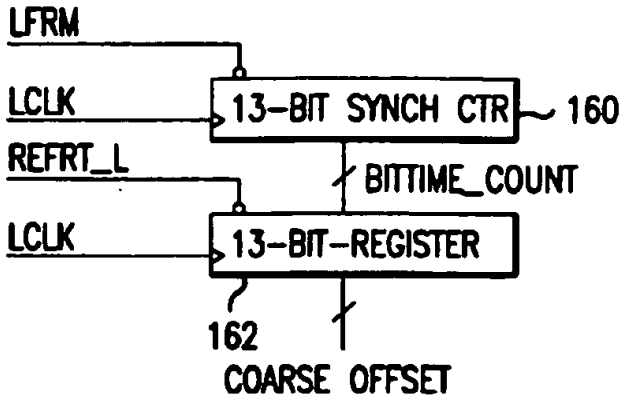

LFRM
LCLK
13-BIT SYNCH CTR
160
REFRT_L
BITTIME_COUNT
LCLK
13-BIT-REGISTER
162
COARSE OFFSET

FIG.7A

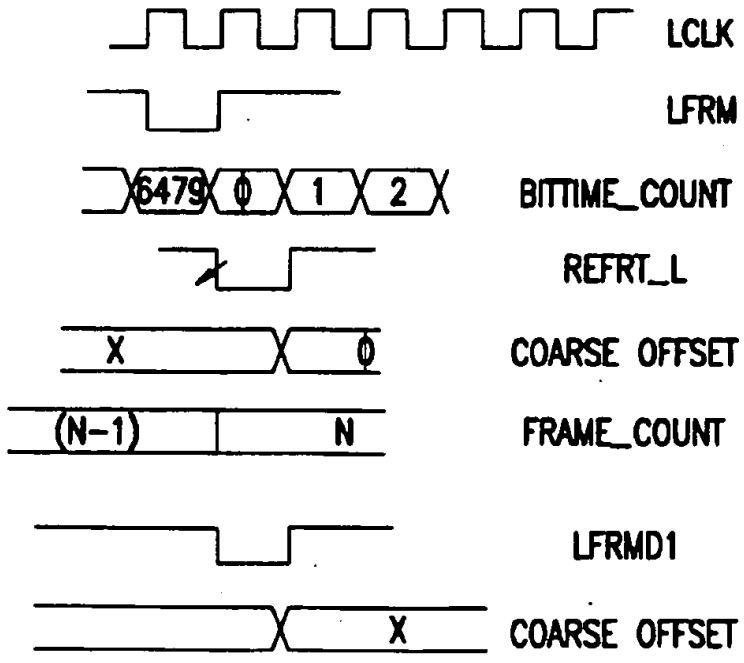

LCLK
LFRM
6479
0
1
2
BITTIME_COUNT
REFRT_L
X
0
COARSE OFFSET
(N-1)
N
FRAME_COUNT
LFRMD1
X
COARSE OFFSET

FIG.7B

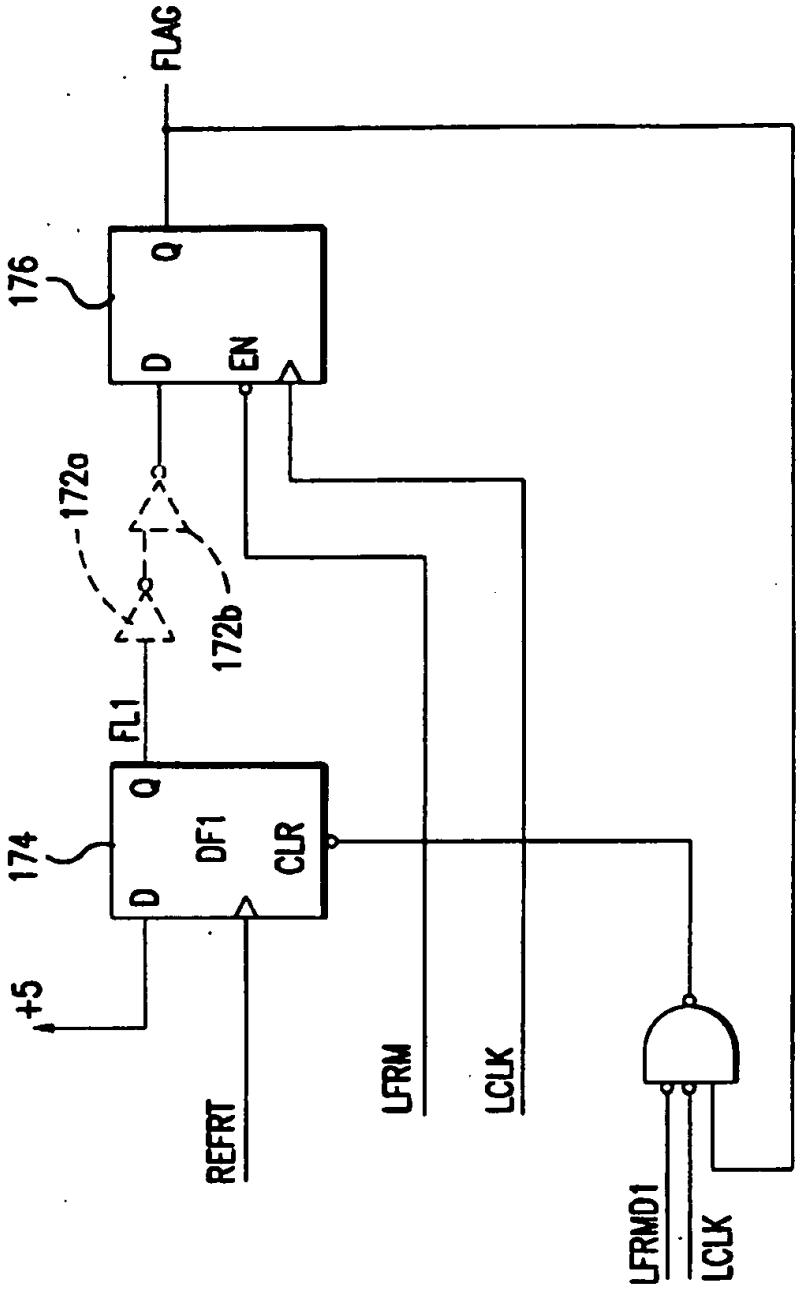

FLAG
176
Q
D
EN
172a
172b
FL1
174
Q
DF1
CLR
D
+5
REFRT
LFRM
LCLK
LFRMD1
LCLK
170

FIG.8

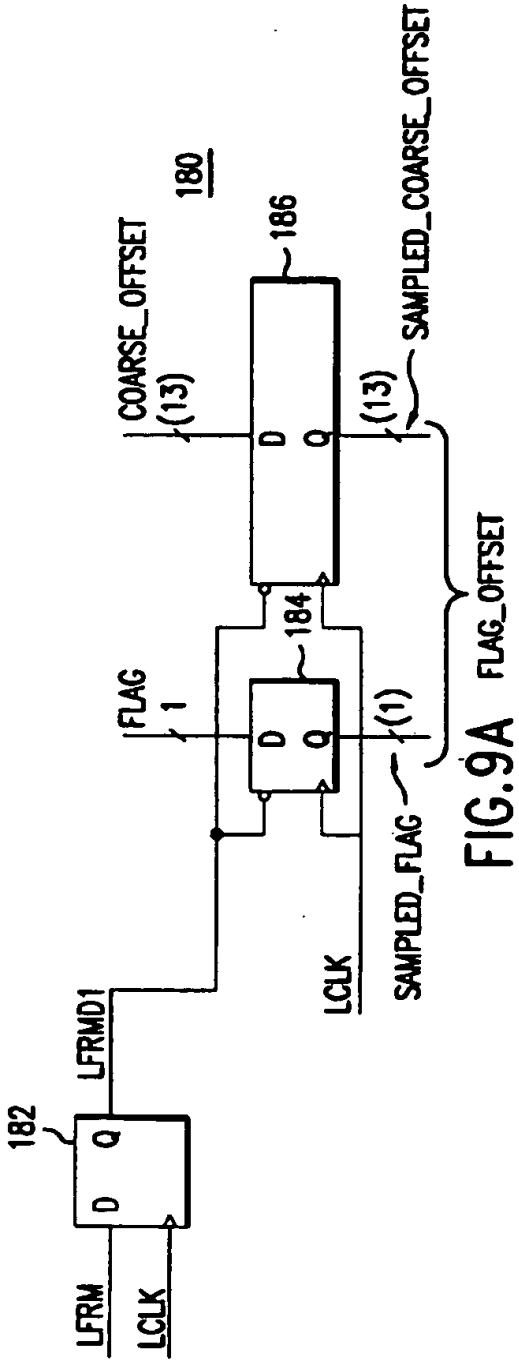

180
COARSE_OFFSET
(13)
186
SAMPLED_COARSE_OFFSET
(13)
FLAG
1
184
(1)
SAMPLED_FLAG
FLAG_OFFSET
182
LFRMD1
LFRM
LCLK
LCLK

FIG.9A

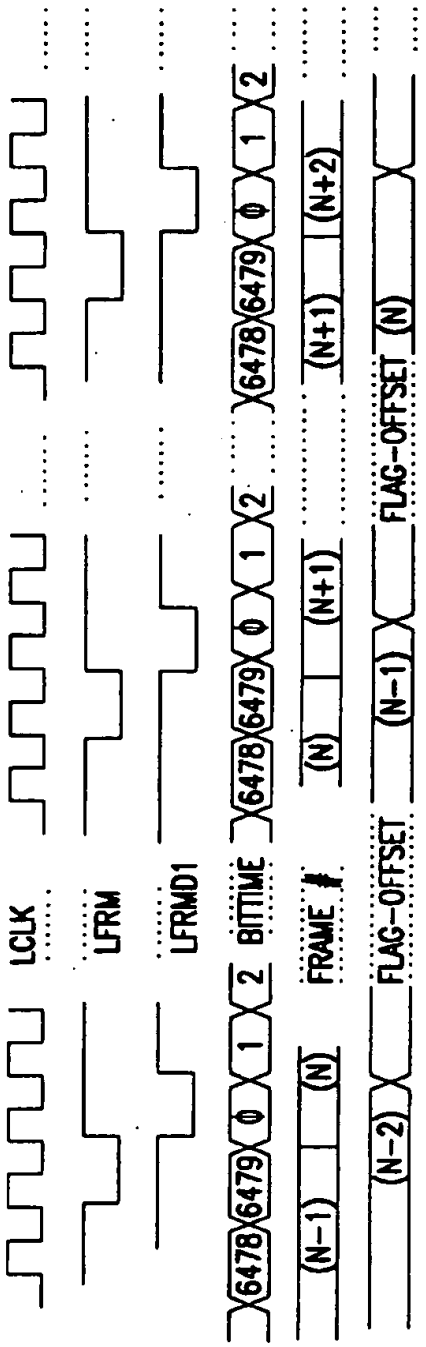

LCLK
LFRM
LFRMD1
6478
6479
0
1
2
BITTIME
(N-1)
(N)
FRAME #
(N-2)
FLAG-OFFSET
(N-1)
(N)
(N+1)
(N+2)
FLAG-OFFSET

FIG.9B

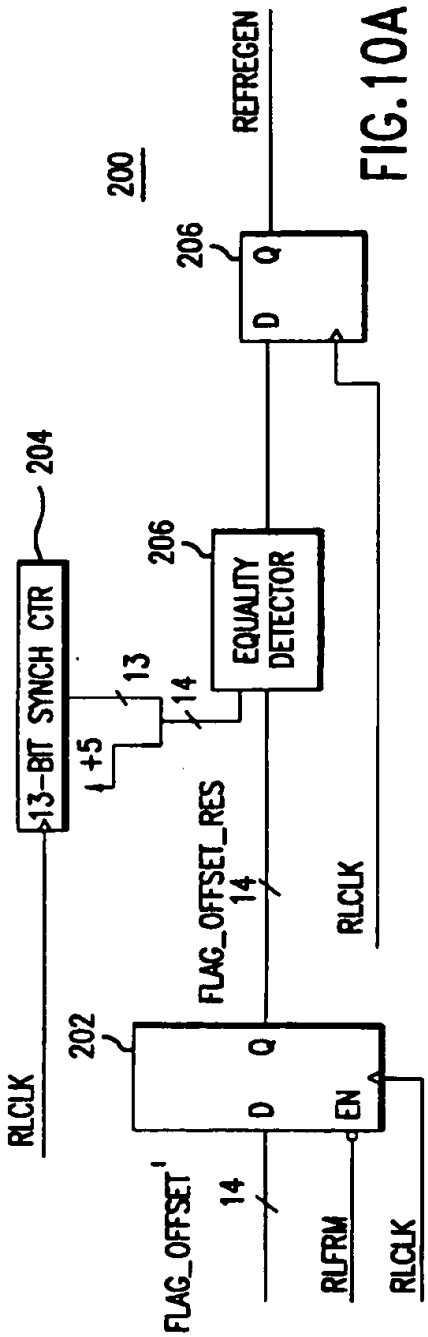


FIG.10A

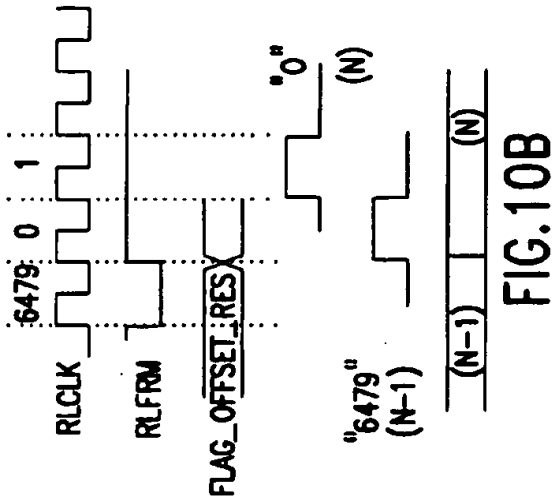


FIG.10B

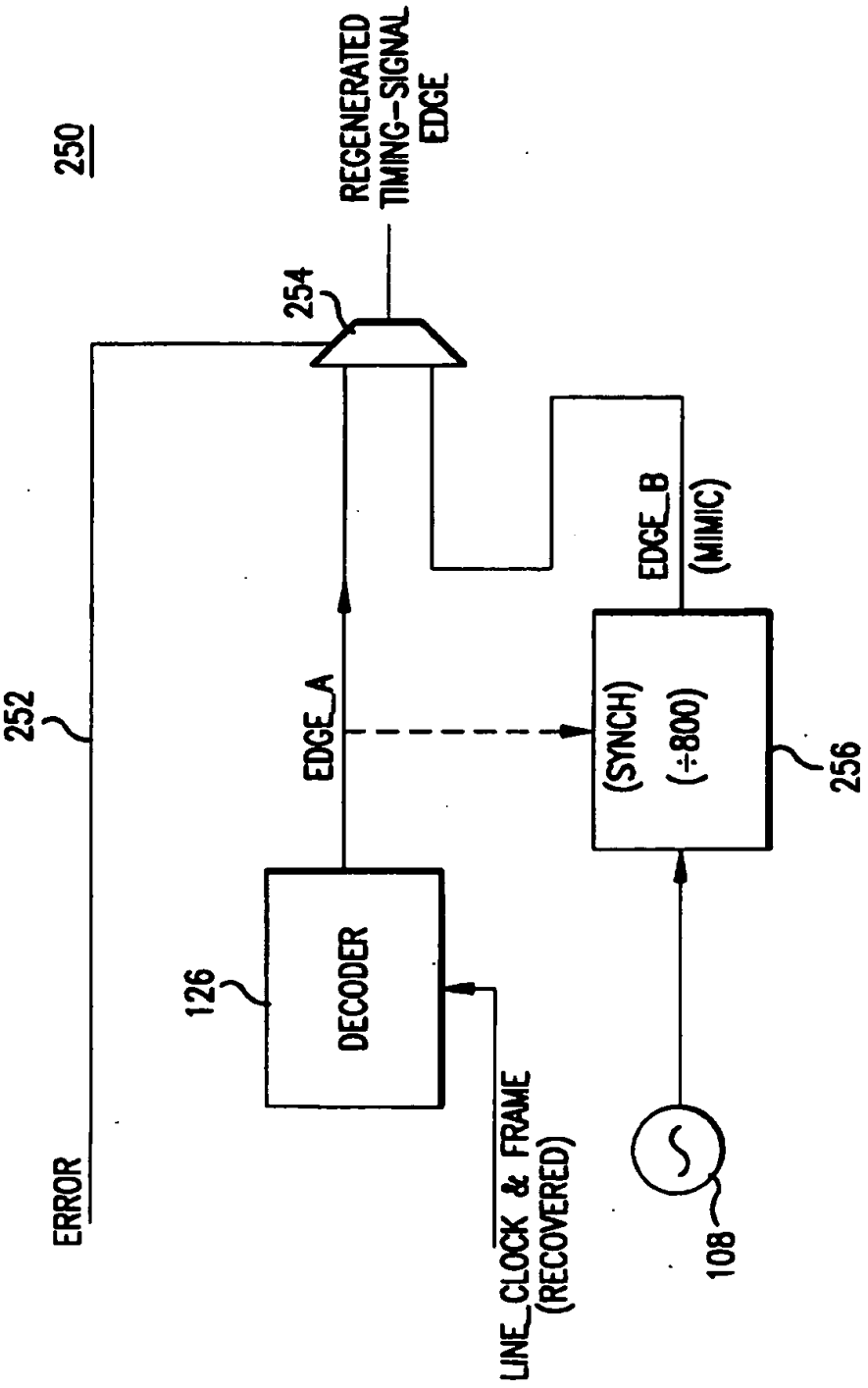

250
REGENERATED TIMING-SIGNAL EDGE
254
252
EDGE_A
EDGE_B (MIMIC)
(SYNCH) (÷800)
256
126
DECODER
ERROR
LINE_CLOCK & FRAME (RECOVERED)
108

FIG. 11

DISTRIBUTION OF SYNCHRONIZATION IN A SYNCHRONOUS OPTICAL ENVIRONMENT

BACKGROUND OF THE INVENTION

1. Field of the Invention

This invention relates to providing synchronization distribution throughout a network and in particular relates to providing this distribution in a synchronous optical communications environment prevalent in telephone networks.

2. Description of the Environment

In digital telephone networks, the network is comprised of hundreds or even thousands of offices or nodes such as shown in simplified form in FIG. 1. The network **10** has a plurality of offices **12, 14, 16, 18, 20**. Each node has a local timing source **12***a*, **14***a*, **16***a*, **18***a*, **20***a*, commonly called a BITS for Building Integrated Timing Supply. Also, each node has a variety of equipment such as switches, optical multiplexers, channel blanks, etc. commonly referred to as network elements (NE's) **12***b*, **14***b*, **16***b*, **18***b*, **20***b*, with the timing for each network element within the office being supplied by the office's BITS. The various offices within the network are connected by copper or optical fiber links **22** called facilities. Unlike the earlier versions of the copper based networks where the facilities formed a mesh type network with each office being linked to multiple offices by several facilities, digital optical networks are arranged in chains or rings with facilities tying each office typically to the two adjacent offices.

Further, in a typical digital network, there are a plurality of primary reference source clocks called PRS clocks. Typically, the PRS clocks are implemented using cesium beam or GPS receiver technology. The PRS clocks serve as master clocks and provide a timing reference for the remainder of the network. The PRS timing is communicated over the facilities to different nodes to permit synchronization between various nodes within the network.

The earlier (non-standard) versions of the optical fiber network employed asynchronous bit stuffing techniques to multiplex the input tributary signals onto the optical line. The distribution of the timing reference in such a network may be realized using an embedded DS1 signal, as shown in FIG. 2. The PRS timing **30** in an office **32** is provided to the BITS **35** and then to fiber multiplexer **36** in a first office and communicated to the next office **40** over the embedded DS1 signal in the optical facility **38**. Further, the fiber multiplexer **42** at the next office **40** recovers the DS1 clock **44** and passes that recovered clock to the BITS **46** of the second office and to the fiber multiplexer **48** for transmission over the next facility in the chain to the next office. Since the BITS clock **46** is not used for generating the line timing signal **50** provided to the next office in the chain (not shown), inaccuracies in the timing reference communicated to the BITS timing in intervening offices do not effect the timing reference communicated to the BITS of the successive offices. Thus, if the BITS timing reference in the second office malfunctions, the synchronization of the successive nodes or offices (not shown) in the network is unaffected. Therefore, each of the nodes or offices in the network may be thought of receiving its synchronization timing directly from the offices containing the PRS. Where each of the nodes of the network is receiving the timing reference directly from the PRS, synchronization may be thought of as being at the same level. Such distribution schemes of synchronization are referred to as being flat.

Although the method described above yields the desirable flat synchronization distribution system, it is not deployed extensively in the telephone network for two reasons. First, the bit stuffing operation performed at each node adds jitter to the embedded DS1 synchronization reference. This may render the DS1 signal unusable(as a timing reference after it traverses a few nodes. Second, and more important, the nonstandard asynchronous optical fiber systems are being replaced by the recently developed standard synchronous optical network technologies, designated as SONET or SDH. The method of distributing the synchronization reference using an embedded DS1 signal does not work properly in the SONET environment, as explained below.

In a SONET multiplexer, the output optical line clock is normally synchronized to the office BITS clock. The rate variations between the input tributaries and the output line signal are accommodated by a byte stuffing process known as pointer adjustment. The eight bit phase movements caused by the pointer adjustments can be large enough to render the embedded DS1 timing reference incapable of adequately transporting the synchronization information. Hence, the standards organizations (ANSI and the ITU) recommend that a DS1 signal embedded within a SONET line signal not be used for synchronization distribution. Instead they recommend the use of the recovered optical line clock to generate a derived DS1 synchronization signal. This derived DS1 signal serves as the synchronization reference input to the office BITS clock.

The use of the derived DS1 to distribute synchronization references, however, implies a hierarchical synchronization network. In such a network, the BITS clock at an intermediate node is not synchronized directly to the PRS but is instead synchronized to the timing reference supplied by the BITS clock in the previous node. This hierarchical scheme for the distribution of synchronization signals has marry shortcomings.

First, administrative controls are required to ensure that a higher quality BITS clock (lower stratum number) does not accept timing from a lower quality BITS clock. Second, the cascade of clocks created by the hierarchical chain can impair the timing reference traversing the network. Third, if a BITS clock fails anywhere in the chain, all the downstream clocks will lose synchronization. And finally, this scheme is prone to the inadvertent creation of timing loops, especially under facility failure conditions. (A timing loop occurs when timing from a first node is passed to the second node and then the timing is fed back through a chain of one or more additional nodes to the first node so that the first node is synchronizing its timing to itself. Such a situation is clearly undesirable since all the nodes involved in the timing loop will be isolated from the PRS).

Synchronization messaging is a solution recommended by the standards organizations to alleviate some of the shortcomings delineated above. In this method, the status of the clock that generates the timing reference at a particular node is communicated to the clocks and network elements at other nodes over a messaging channel. The clocks at these other nodes will then decide, in an intelligent manner, whether they should synchronize to one of the incoming timing references, or whether they should operate in a holdover mode. However, the synchronization messaging scheme does not cure all the problems created by the hierarchical synchronization distribution network. Furthermore, implementation of this messaging scheme will be expensive as it involves the retrofitting of the existing BITS clocks and the SONET network elements to provide this capability.

Therefore, it is the first objective of this invention to provide a method for transporting network synchronization

reference signals over the existing SONET network using a flat distribution scheme. It is a second objective of this invention to distribute these synchronization reference signals without incurring the problems associated with the hierarchical scheme. It is yet a third objective of this invention to achieve the flat synchronization distribution system without requiring substantial hardware investment or retrofitting costs.

SUMMARY OF THE INVENTION

These and other objects are achieved by relying on two timing elements available at each office: the line clock and the SONET frame timing. A timing reference signal synchronized to the PRS and at a frequency that is at least slightly less than the frame rate is generated at the originating PRS site. The line clock is then used to determine the interval between the start of the frame and an edge of the timing reference signal. This timing difference is encoded and transmitted in an overhead channel and may be decoded at the next node. The next node may then recover the timing reference for use in its own BITS timing and for transmission on to the next node.

Therefore, a flat synchronization structure is created as each node in the network depends for its timing upon the original PRS instead of the intervening nodes. Further, this flat structure eliminates any possibility of timing loops.

To fit this approach in existing network structures without substantial hardware costs, a few counters, flip-flops, and gates can be used to generate all of the timing signals. To reduce messaging overhead, the encoded timing difference can be transmitted over multiple frames in currently used control bytes reserved in the SONET architecture.

DESCRIPTION OF THE DRAWINGS

FIG. **1** is a diagram of a simple prior art telephone network.

FIG. **2** is a diagram of the synchronization distribution scheme in a prior art asynchronous network.

FIG. **3** is a block diagram for an encoder and a decoder for an embodiment of the invention.

FIGS. **4A** and **4B** are timing diagrams relating to the embodiment of FIG. **3**.

FIG. **5** is a functional flow diagram for the embodiment of FIG. **3**.

FIGS. **6A** and **6***b* are schematics of circuits for retiming the timing reference signal at the encoder.

FIG. **7A** is a schematic of a circuit in the encoder for measuring the timing difference between the start of the frame and the timing reference signal.

FIG. **7B** is the timing diagram for FIG. **7A**.

FIG. **8** is a schematic of a circuit in the encoder for generating a flag to indicate in which frame an edge of the timing reference signal has occurred.

FIG. **9A** shows a circuit in the encoder for sampling the measured timing difference and the flag.

FIG. **9B** is the timing diagram for FIG. **9A**.

FIG. **10A** is a circuit in the decoder for generating the timing reference signal in the decoder.

FIG. **10B** is the timing diagram for FIG. **10A**.

FIG. **11** is a circuit for generating a substitute for the timing reference signal when certain error conditions are detected.

DETAILED DESCRIPTION OF THE INVENTION

The embodiments of the invention involve transmission of synchronization of timing. FIG. **3** is useful in explaining an embodiment of the invention, as applied in a SONET or similar synchronous optical environment. A portion **100** of the network is shown in FIG. **3**. Each node or office **102**, **104**, in the network has a BITS clock source **106**, **108** and at least one node has a PRS **110** directly controlling the BITS timing **106** at the same office **102**. Each node receives and transmits a line clock LCLK **114** over a facility such as an optical fiber **112** linking nodes. Each node also receives and transmits frames at a nominal rate of eight thousand times a second with the frame containing control information and data according to the established network protocol. In the SONET environment, the duration of a frame is nominally one hundred twenty five microseconds. A locally generated timing reference signal **116** generated from the BITS timing signal (the BITS timing signal is nominally a 1.544 MHz signal in SONET) is also provided in each node having a PRS. Also generated internally by the office or node **102** is a frame start signal LFRM **122**. The frame start signal and the line clock can be obtained from the add-drop multiplexer (ADM, not shown) at the office.

An encoder **120** measures the difference in timing between the start of the frame as indicated by the frame start signal, signal LFRM **122** and the timing reference signal **116**. This timing difference may be obtained by counting the line clock pulses between the start of the frame as indicated by LFRM **122** and a pulse edge of the locally generated timing reference signal. The timing difference represented by this count may be encoded into reserved control bytes of the message frame **115** and may then be transmitted over the facility **112** to the next office **104** in the SONET chain or ring. At this second office, a decoder **126** uses the transmitted line clock **114'** that has been transmitted from the office **102** and recovered in the office **104** by the ADM (not shown) along with a line frame start signal **122'** that has been reconstructed by the office ADM according to well known techniques in the art. In a manner explained below, the local timing reference signal **116'** can be regenerated, for example, by multiplying the period of the line clock LCLK **114** by the transmitted count **115** to generate a pulse in a manner that will be described in more detail below.

That regenerated timing signal **116'** may then be supplied to a further encoder **120'** that also receives the start of the frame signal **124** generated by the office for the frames to be transmitted. The line clock **114"** for transmission to the next node in the chain of the network (not shown) is also provided to the encoder **120'** from the ADM (not shown). The difference between the start of the frame pulse LFRM **124** and the regenerated timing signal **116'** may be counted with the line clock to provide a further count **115'** for transmission over the facility (not shown).

FIG. **4A** shows a timing chart relevant to time measurement at the first office **102**. The 51.84 MHz line clock LCLK **114** provides the fundamental reference for counting periods or bit times for the time measurement. The start of each successive frames N−1, N, N+1, N+2 is indicated by the rising edge of a pulse in the frame start signal LFRM **122**. Since a frame has a duration of 125 microseconds, there are 6480 possible periods of the line clock LCLK in which an edge of the timing reference signal **116** can occur. In the instance in frame N, the edge occurs during the fourth bit time measured in units of the 51.84 MHz clock so a count of four would be encoded. During the next frame (N+1) or some subsequent frame, that count may be transmitted to the next node **104** over the link **112**.

Upon receipt of the count, the next node **104** in the network will generate a rising edge of a reconstructed timing reference signal **116'** at the start of the fourth received bit

time in the N+K frame as shown in FIG. 4B. In particular, the decoder **126** will receive the regenerated line clock LCLK **114'** generated by the ADM at the office **104** and count a number of cycles of that clock equal to the received count. At that point, the decoder will generate an edge in a regenerated timing reference signal **116'** that may be used in the office **104** for synchronizing the BITS **108** to the PRS **102**. In addition, the line clock, the regenerated timing signal **116'** and the local frame start signal **124** from the second office's **104** SONET ADM (not shown) may be used for measuring the difference and transmitting a count to the next office so that it may also generate a local version of the timing signal. Since each version of the timing signal is only dependent upon the PRS timing and not the BITS timing of each local office in the chain, distribution of the timing tied to the timing reference is flat and avoids both timing loops and hierarchy problems.

FIG. **5** shows a functional flow diagram of distribution of timing throughout the network. Successive offices N−1, N and N+1 in a network perform the function of measuring the timing difference between a timing reference signal and the line clock and the start of the frame using the transmit line clock. The measured time difference at node N−1 is encoded and then transmitted over a link **150**. At the next node, or office N, the line clock and the frame timing are recovered, the transmitted count is decoded and used to regenerate the timing reference signal. This regenerated timing reference signal is supplied to the BITS at office N for synchronization. This regenerated timing reference signal at office N is also supplied for measuring the timing difference with the start of the frame at office N to be transmitted to the next node using the transmit line clock and frame timing of office N. The timing difference measured at office N is encoded and transmitted over the facility for subsequent recovery, decoding and regeneration at office N+1 in a like manner.

Selection of the appropriate criteria for the local timing reference signal is based upon several factors. First, the local timing signal pulse edge should occur no more than once each frame. Therefore, the local timing signal should have a rate of less than or equal to the frequency of a frame; e.g., 8 KHz in a SONET network. However, since the frame rate may be at a slightly lower rate and still be within the SONET specification, the local timing reference signal should preferably occur at a rate that is less than the minimum permitted frequency of the frames. Further, the frame rate and the timing signal rate should not be harmonics of each other. Optimally, they should be "as prime as possible" with respect to the measurement rate of the line clock, which is 51.84 MHz. In particular, the highest common factor of the frame rate of 8 KHz and the timing reference signal should be as low as possible to prevent the development of beats occurring with such sampling with the line clock. In addition, the timing rate signal should preferably be at a frequency readily obtainable from frequencies in the office such as the BITS signal. For this reason, a timing rate of 7.72 KHz or an integral submultiple thereof (i.e., 3.86 KHz, 1.93 KHz, 0.965 KHz) are among the preferred frequencies for a SONET network. These frequencies can be readily generated from the 1.544 MHz BITS timing signal available in each office since 7.72 KHz is obtained readily by dividing the BITS signal by two hundred. In fact, for reasons discussed later, 1.93 KHz may preferably be used for optimal encoding in an overhead byte for transmission over a SONET network.

A further advantage of the 7.72 KHz timing signal or some integral submultiple thereof is avoidance of metastable states resulting from the synchronous nature of the various signals and the switching speed of the digital logic involved. In particular, with the line clock of 51.84 MHz, there is a window around each edge of the line clock in which the occurrence of an edge of the timing signal may not be detected due to the transistor switching delays inherent in the digital logic. Such metastable conditions would result in a delay in detecting the edge and hence inject a one clock period pulse offset in detecting the count representing the occurrence of the timing edge. If a signal having the same frequency as the frame rate, or harmonic thereof, is selected, this metastable condition can persist over a substantial period of time. With the selection of 7.72 KHz or an integer submultiple of that frequency, any metastable events will be one time events that can readily be eliminated at the next office through the use of a phase lock loop in the generation of the timing reference signal. By use of 7.72 KHz or integer submultiples of 7.72 KHz, the edge to edge change in the timing reference signal ensures that if a timing signal edge occurs in the metastable region of the edge in the line clock for the logic, the next occurrence of the timing reference signal edge will not be during the metastable region.

FIGS. **6A** through **11** show various circuitry and associated timing diagrams for generating and regenerating the timing reference signal at the various nodes throughout the network. It is assumed that in each of these circuits, the circuit components are synchronous. First, the timing signal REF should preferably be retimed to the timing of the line clock LCLK. FIG. **6A** shows a circuit for generating such retiming while reducing the likelihood of a metastable condition. The circuit comprises three edge triggered D flip-flops and the retimed timing signal output is REFRT and its complement, REFRT_L. FIG. **6B** shows an alternative version of such a circuit.

FIG. **7A** shows a circuit for generating the count representing the timing difference in units of the line clock LCLK period between the pulse indicating the start of the frame N in signal LFRM, and the retimed timing reference signal. The rising edge of the start of the frame signal LFRM resets a thirteen bit counter **160** that counts the line clock LCLK. When the falling edge of the complementary retimed timing signal pulse RFRT_L occurs, it enables the input of a thirteen bit register **162** that is coupled to the output BIT TIME COUNT of the synch counter **160** and the line clock LCLK. The current value from the counter is clocked at this point into the shift register. The contents of this register, labelled COARSE OFFSET, represent the timing delay between the LFRM frame pulse and the retimed timing signal in units of time defined by the LCLK period. The contents of the counter are held in the register until the next edge of the retimed timing signal. FIG. **7B** depicts the associated timing diagram.

Since the retimed timing reference signal may not have an edge in each frame, a flag indicating when an edge has occurred is needed. FIG. **8** shows a circuit that is useful for generating the flag signal to indicate that an edge in RFRT_L occurred during the current frame. The circuit **170** receives the line clock LCLK, the retimed timing reference signal REFRT, the start of the frame signal LFRM, and a start of the frame signal delayed by one line clock period LFRMD1. Optionally, two inverters **172***a* and **172***b* may be provided between D flip flops **174** and **176** that generate a flag signal FLAG indicating an edge of the timing reference signal has occurred with a true value indicating a flag has occurred.

For subsequent transmission to the next node, the stored count must be sampled for encoding and transmitting. The flag and the coarse offset value are then stored as a fourteen

bit word as shown in FIG. 9A. The frame start timing signal LFRM is delayed by the line clock LCLK in a flip flop 182 to provide the LFRMD1 signal and that LFRMD1 signal enables a D flip-flop 184 and a register 186 circuit that receive as their inputs the FLAG and COARSE_OFFSET values. These two values are gated with the line clock LCLK for processing during the subsequent frame to provide the sampled fourteen bit entity FLAG_OFFSET. The FLAG_OFFSET comprises SAMPLED_FLAG and SAMPLED_COARSE_OFFSET. As shown in the timing diagram FIG. 9B, the FLAG_OFFSET value lags one frame behind where the edge of the retimed timing reference signal RFRT occurs (assuming an edge occurred during the prior frame).

If the resultant sampled coarse offset value has the sampled flag high, that count value may then be processed by the office network element, such as an ADM, for transmission over the network to another node on the network according to the network protocol. For example, using the current SONET protocol, the network has a overhead byte called the F1 byte that is unused and reserved for future applications. Therefore, it is possible to use the F1 byte for transmitting the synchronization information.

Given that the synchronization count (COARSE_OFFSET) uses the SONET standard of an 8 KHz frame rate and a line clock of 51.84 MHz, the coarse offset requires thirteen bits to transmit the maximum possible count of 6479. Therefore, at a minimum, two F1 bytes in two separate frames may be used for transmitting the information.

However, for coding accuracy, it is more desirable to transmit the information over four frames and therefore use four F1 bytes to permit error detection. Therefore, to match this transmission rate of four frames, the frequency of the timing reference signal should be 1.93 KHz or some integral submultiple of that rate. The timing edge that occurs in Frame N, is actually encoded and transmitted in a four frame sequence over the next four frames, N+1, N+2, N+3, and N+4. Regeneration of the edge at the receiving node when using this algorithm will occur at a minimum of five frames after the edge occurred at the transmitting node.

A possible format for the F1 bytes is shown in table I below:

Byte No.	MSB	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	LSB
0	0	$Edge_1$	$Edge_2$	Coarse Offset 5 LSB				
1	0	$Edge_1$	$Edge_2$	Coarse Offset Next 5 Bits				
2	0	$Edge_1$	$Edge_2$	Coarse Offset 3 MSB			X	X
3	1	X	X	CRC	CRC	CRC	CRC	CRC

According to this format, the most significant bit in each of the first three F1 bytes is set to a logical zero and the most significant bit for the last byte is set to 1 so that the last byte of a four frame sequence can be readily detected. Alternatively, the MSB of the first byte may be set as one to mark the start of a sequence and the three remaining bytes may be set to zero. Further, the fifth and sixth bits in the first three F1 bytes of a four frame sequence are called edge data and are used for determining during which frame of the prior four frame sequence an edge of the retimed timing reference signal occurred.

In this implementation, the receiving node office compares the edge data in each of the first three F1 bytes of a four frame sequence. Either a majority rules or a requirement of all three edge bit patterns being the same may be used to determine in which frame the edge occurred. Transmitting different values for each of $Edge_1$ and $Edge_2$ for each of the first three frames for the four frame sequence can be used to indicate that no edge occurs during the four frame sequence, a link has been broken so synchronization to the PRS has been lost or a phase slippage greater than the predicted amount has occurred. In addition, various such error conditions can be passed between the nodes by encoding such error conditions in the bits labelled with X.

The value of the coarse offset is encoded into the first three F1 bytes of the four frame sequence. Also, a cyclic redundancy check (CRC) sequence, or some other error detection mechanism, is transmitted in the F1 byte of the fourth frame for detecting transmission errors in the transmitted Coarse Offset.

The bits labelled with X may be used for a variety of optional functions. For example, such bits may be used with the bits indicated as CRC for transmitting a nine bit error correction code instead of a CRC. A predetermined bit pattern for these bits may also indicate a phase slippage at the transmitting node that is greater than a predetermined threshold or the like to prevent propagation of phase errors. Detection of such errors is readily possible by either the transmitting or receiving office. In response to such conditions, the receiving office may disregard the transmitted count for a four byte sequence.

For example, the receiving office may compute an expected range of values for the received coarse offset based upon the relative frequency of the frame and the timing reference signal. The receiving office may disregard the received coarse offset if the value is beyond the expected range.

At the receiving node where the timing difference information (the encoded coarse offset count) is received, the timing difference information can be used to regenerate a timing reference signal with the use of the circuit 200 shown in FIG. 10A. In particular, the regenerated received and recovered line clock RLCLK and the regenerated received frame signal RLFRM are provided from the receiving office ADM (not shown). The transmitted F1 bytes are decoded by the receiving office in the chain to provide a reconstructed version of FLAG_OFFSET. This regenerated flag signal can be based upon the edge bits in each of the first three F1 bytes in the transmitted four frame sequence. Either a majority rules protocol or a requirement of all three sets of edge bit patterns matching can be used for determining in which frame the edge occurred. To match the flag signal at the equality detector, a logical "1" is also provided so that the pulse will be generated during the appropriate frame. The coarse offset can be obtained by concatenating the three portions of the coarse offset transmitted in the four-frame sequence.

The decoded FLAG_OFFSET is clocked into a register 202 that is enabled by the reconstructed start frame signal. Simultaneously, a thirteen bit counter 204 is counting the regenerated line clock RLCLK. Both the output of the counter 200 and the register 202 are provided to an equality detector 206. The equality detector can then provide a pulse when the count contents and the "one" to match the flag and the contents of the register are equal.

The pulse from the output of the equality detector can then be provided to a D flip flop 208 to provide the regenerated timing reference signal REFREGEN. This regenerated timing signal is provided at least two frames plus one regenerated line clock period after the timing edge occurred. If the format described above using four frames for transmission of the timing difference is used, the delay will be at least five

frames plus the one regenerated line clock period. Any other delays inherent in communications between the two offices may increase that delay. Nonetheless, such a regenerated reference timing signal does permit transmission of synchronization throughout a network in the manner described for FIGS. 3–5 above.

Further, although the use of the line clock at 51.84 MHZ provides a granularity of about twenty nanoseconds (the period of the line clock) with which to measure synchronization, this granularity can be reduced. To reduce the granularity, one may provide the recovered timing reference signal REFREGEN to a digital phase lock loop with a very narrow bandwidth, for example about one hertz. The granularity of the sampling of the frequency with the line clock results in a poisson like distribution of error in the sampling of the phase relationship between the timing signal and the start of the frame pulse. A narrow bandwidth filter over the long term filters out virtually all of the phase error due to this poisson like distribution arising from the granularity assuming the PRS signal is highly stable; i.e., maintaining an accuracy in one part of 10^{13} over the course of a day. Thus, the use of such narrow bandwidth phase lock loops results in a much more tightly controlled synchronization once the phase lock loop has stabilized.

In particular, with a line clock at about a period of twenty nanoseconds, the best phase synchronization that could be attained would be on the order of twenty nanoseconds. However, by employing a one hertz bandwidth phase lock loop the phase error over the long term can be reduced to about one percent of this granularity on or about the order of 0.2 nanoseconds. This phase lock loop can also be used for generating the 1.544 MHz timing reference signal required by the BITS clock at the node using a standard frequency multiplier configuration.

Further, to achieve phase lock during start up or after various transient conditions, it is preferable that the bandwidth of the filter be adaptive as is readily possible with digital filters. During start up or after various transient conditions, the bandwidth of the loop is opened up, permitting faster acquisition of phase lock.

Upon detection of errors in either the CRC or upon detection of the predetermined bit pattern in any of the bits indicated with an X in Table 1 above indicating an error condition, the receiving node can go into a holdover node. Also, for such holdover conditions, the regenerated value can be ignored by the phase lock loop and the system can resort to the predicted values that may be generated by knowing the frequency of the frame rate and the timing reference signal. Alternatively, with such phase lock loops, the loop can be held at a nominal frequency until the cause of the holdover condition is alleviated.

Also, the BITS clock can also be used for providing the timing signal temporarily when there is an error condition in the encoded coarse offset information, or the edge information, as shown in FIG. **11**. If an error is detected by the receiving office in the CRC, the edge bits, or a sudden change in the coarse offset value from a predicted value shows a loss of synchronization at the transmitting office, the office generates an error signal **252**. The output of the decoder **126** is provided to one input of a multiplexer **254**. The other input is provided by dividing down the BITS clock **108** by eight hundred with divider **256** where the output of the divider is synchronized to valid rising edges of the regenerated signal according to well known techniques. The output of the divider **256** is coupled to the other input of the multiplexer **254** to provide a temporary backup version of the regenerated timing signal. Therefore, whenever the office detects an error condition, the error signals **252** can select the temporary back up signal to provide the regenerated timing signal **114**.

For propagation of synchronization throughout the network, each node not serving as a master clock source can both receive and regenerate the timing reference signal and also generate a timing reference signal and transmit the difference between that generated timing reference signal and the frame. Since each node in the network receives and regenerates the same synchronization information, the network architecture is essentially flat. Further, timing loops are eliminated due to the usage of such a flat architecture.

Although a particular embodiment of the invention is disclosed, alternatives would be readily apparent to those of skill in the field. Different frequencies for signals are, of course, appropriate for different networks such as OC-N or SDH where line clock frequencies are integer multiples of 51.84 MHz or 155 MHz, respectively. In fact, a 19.44 MHz clock, which is readily available in many implementations, may also be used, instead of the actual line clock, to measure the timing difference. Also, different protocols can be used for encoding and transmitting the timing difference signal. Instead of using counters to generate the timing difference various types of analog and digital phase detectors may be used. Alternatively, the regenerated timing signal could also be obtained through the use of a high precision numerically controlled oscillator controlled by a microprocessor using the coarse offset information to generate the timing reference signal at the output of the oscillator. In addition, while the disclosed embodiments use the start of the frame as a reference for generating the timing difference, other specific times in the frame may also be used for generating the timing difference with the local timing reference signal.

Further, instead of using the PRS as an original source for the synchronization signal to be distributed, other sources may be used such as the disciplined time scale generator disclosed in U.S. patent application Ser. No. 08/278,423 to Zampetti, pages, 9–26 of which are incorporated herein by reference. By equipping occasional offices in the chain with such timescale generators that are disciplined to a universal time scale such as GPS or LORAN, a highly synchronous network is established without requiring the expense of numerous PRS clocks or disciplined time scale generators at each office. Resort to the scope of the invention should be through the claims.

We claim:

1. A method of passing synchronization through a network comprised of a plurality of nodes communicating with each other at a predetermined frame rate, the nodes communicating with each other through frames having predefined starts and with a line clock, the communication occurring by transmitting the frame and the line clock between nodes, the method comprising:

generating at a first node a local timing reference signal at a frequency that is less than the frame rate;

determining with the line clock at the first node the timing difference between a predetermined time of the frame and the local timing reference signal; and

transmitting the timing difference to at least one other node in the network.

2. The method of claim 1, wherein the method further comprises:

determining the time of the frame and recovering the line clock at a second node coupled to the first node;

regenerating the timing reference signal based upon the transmitted timing difference.

3. The method of claim 1, wherein the frequency of the local timing reference signal is selected to minimize the occurrence of the metastable states in successive cycles of the timing signal.

4. The method of claim 1, wherein the net work is a SONET network and the SONET network frame includes an F1 byte, the transmission of information indicating each timing difference occurs over multiple frames in the F1 byte.

5. A method for maintaining synchronization in at least a portion of a network of a plurality of nodes, the network having a reference timing signal, each node in the network communicating with at least one other node in the network through frames having frame timing and each node generating a line clock and frame timing for communication with at least one other node, the method comprising:

at at least one first node generating a local timing reference signal based at least in part upon the network reference timing signal;

measuring the difference in timing at the first node between the frame timing for at least some frames being transmitted and the local timing reference signal;

transmitting in at least some frames to at least one second node from the first node the measured time difference; and

generating at the second node with the line clock, the frame timing and the transmitted measured time difference a reconstruction of the local timing reference signal such that at least one reconstructed local timing reference signal is synchronized to the network reference timing signal.

6. An apparatus for aiding the distribution of synchronization throughout a network comprising a plurality of nodes, each node generating a line clock having clock pulses at a predetermined frequency for transmitting information to another node and each node transmitting according to frames having a period based upon the timing of the frame that is being transmitted, the apparatus further including:

a clock generator providing a timing reference signal having a period that is greater than the period of the frame rate; and

a timing difference detector detecting a timing difference from time to time between a specific time in the frame and the timing reference signal.

7. The apparatus of claim 6, wherein the detected timing difference is transmitted to another node within the network.

8. The apparatus of claim 6, wherein the timing difference is detected by counting the number of clock pulses between a fixed reference point in the timing reference signal and the start of a frame.

9. The apparatus of claim 8, wherein the frequency of the timing reference signal is relatively prime when compared with the frame timing and the line clock pulse rate.

10. The apparatus of claim 6, wherein the frame timing has a frequency of about 8 KHz and the timing reference signal has a frequency of about 7.72 KHz or an integral submultiple of 7.72 KHz.

11. The apparatus of claim 6, wherein the node also regenerates a line clock signal received from a second node and regenerates a frame timing received from frames received from said second node, and the apparatus receives information from time to time indicating the difference in time between the start of a received frame and a timing reference signal at said second node, the apparatus further including:

means responsive to the received timing difference for regenerating the timing signal at the first node.

12. The apparatus of claim 11, wherein the timing reference signal is generated based upon the regenerated timing signal.

13. The apparatus of claim 11, wherein the regenerated timing signal is the timing reference signal.

14. An apparatus for aiding the distribution of synchronization throughout a network comprised a plurality of nodes, each node regenerating a line clock at a predetermined frequency for receiving information from another node and each node receiving data according to frames having a period based upon the timing of the frame that is being received, the network further including transmission of information from a node related to the timing difference based upon the line clock between a predetermined portion of the frame and a timing reference signal, the apparatus further including:

means for regenerating the line clock;

means for regenerating timing associated with the received frame; and

means for reconstructing the timing reference signal from the received information, the regenerated line clock and the regenerated frame timing.

15. The apparatus of claim 14, wherein the timing difference is detected by counting the number of regenerated clock pulses from the point of the frame timing based upon the received information.

16. The apparatus of claim 15, wherein the frequency of the timing reference signal is a relative prime when compared with the frame timing.

17. The apparatus of claim 14, wherein the frame timing has a frequency of about 8 KHz and the reference timing signal has a frequency of about 7.72 KHz or an integral submultiple of 7.72 KHz.

18. A method for generating synchronization throughout at least a part of a network based upon a master clock, each node of the at least part of the network having a node clock, whereby the phase timing of the node clock is synchronized to the timing of the master clock with network protocol including a predetermined timing relationship for the information being transmitted, the method including:

measuring a timing difference between a reference signal and the predetermined timing relationship for the information being transmitted, the measurement being achieved with a predetermined granularity;

transmitting the measurement to at least one other node where the measurement is received;

synchronizing the clock of the node to the master clock using the received information, the timing relationship of the network and a filtering algorithm to provide a long term phase synchronization with the master clock at a resolution that is at least one order of magnitude smaller than the measuring granularity.

19. The apparatus of claim 18, wherein the means for reconstruction comprises a counter counting a predetermined number of line clock pulses from a predetermined portion of the frame based upon the transmitted information to thereby generate a pulse for the timing reference signal.

20. The method of claim 19, wherein the method further includes transmitting the measurement from a node having a master clock to a second node and the second node generating a timing signal in a phase lock relationship with the master clock based upon the transmitted information, the second node using the generated timing signal for making a timing difference measurement for transmission to a third node, the clock in the third node being maintained in synchronization with the first node based upon the transmitted timing information.

21. A method for transmitting timing information throughout at least part of a network, the network transmitting information during predetermined timing slots and a propagation delay existing in the network in the transmission of information between adjacent nodes, the method including:

making a timing measurement based upon a timing reference and a timing information being transmitted during a first timing slot;

transmitting the timing measurement during a subsequent timing slot;

recovering the timing measurement at a second node during the immediately subsequent timing slot such that the timing information at the second node is recovered two timing slots plus a propagation delay after the first timing slot.

22. A method of passing synchronization through a network comprised of a plurality of nodes communicating with each other at a predetermined frame rate, the nodes communicating with each other through frames having predefined starts and with a line clock, the communication occurring by transmitting the frame and the line clock between nodes, the method comprising:

generating at a first node a local timing reference signal;

determining with the line clock at the first node the timing difference between a predetermined time of the frame being transmitted and the local timing reference signal;

transmitting the timing difference to at least one other node in the network.

23. The method of claim 22, wherein the method further comprises:

determining the time of the frame and recovering the line clock at a second node coupled to the first node;

regenerating the timing reference signal based upon the transmitted timing difference.

24. The method of claim 22, wherein the frequency of the local timing reference signal is selected to minimize the occurrence of the metastable states in successive cycles of the timing signal.

25. The method of claim 22, wherein the network is a SONET network and the SONET network frame includes an F1 byte, the transmission of information indicating each timing difference occurs over multiple frames in the F1 byte.

26. An apparatus for aiding the distribution of synchronization throughout a network comprising a plurality of nodes, each node generating a line clock having clock pulses at a predetermined frequency for transmitting information to another node and each node transmitting according to frames having a period based upon the timing of the frame that is being transmitted, the apparatus further including:

a clock generator providing a timing reference signal; and

a timing difference detector detecting a timing difference from time to time between a specific time in the frame being transmitted and the timing reference signal.

27. The apparatus of claim 26, wherein the detected timing difference is transmitted to another node within the network.

28. The apparatus of claim 26, wherein the timing difference is detected by counting the number of clock pulses between a fixed reference point in the timing reference signal and the start of a frame.

29. The apparatus of claim 28, wherein the frequency of the timing reference signal is relatively prime when compared with the frame timing and the line clock pulse rate.

30. The apparatus of claim 26, wherein the frame timing has a frequency of about 8 KHz and the timing reference signal has a frequency of about 7.72 KHz or an integral submultiple of 7.72 KHz.

31. The apparatus of claim 26, wherein the node also regenerates a line clock signal received from a second node and regenerates a frame timing received from frames received from said second node, and the apparatus receives information from time to time indicating the difference in time between the start is a received frame and a timing reference signal at said second node, the apparatus further including:

means responsive to the received timing difference for regenerating the timing signal at the first node.

32. The apparatus of claim 31, wherein the timing reference signal is generated based upon the regenerated timing signal.

33. The apparatus of claim 31, wherein the regenerated timing signal is the timing reference signal.

* * * * *

UNITED STATES PATENT AND TRADEMARK OFFICE

CERTIFICATE OF CORRECTION

PATENT NO. : 5,828,670
DATED : October 27, 1998
INVENTOR(S) : Narasimha, et al.

It is certified that error appears in the above-identified patent and that said Letters Patent is hereby corrected as shown below:

Claim 4, line 1, replace "net work" (two words) with --network--.

Claim 5, line 8, delete the first occcurence of "at".

Claim 31, line 6, replace "is" with --of--.

Signed and Sealed this

Sixteenth Day of March, 1999

Attest:

Q. TODD DICKINSON

Attesting Officer

Acting Commissioner of Patents and Trademarks